**DOI** 10.15622/ia.23.1.10

B. SWAPNA, V. DIVYA

# LATENCY AWARE INTELLIGENT TASK OFFLOADING SCHEME FOR EDGE-FOG-CLOUD COMPUTING – A REVIEW

*Swapna B., Divya V.* **Latency Aware Intelligent Task Offloading Scheme for Edge-Fog-Cloud Computing – a Review.**

**Abstract.** The huge volume of data produced by IoT procedures needs the processing power and space for storage provided by cloud, edge, and fog computing systems. Each of these ways of computing has benefits as well as drawbacks. Cloud computing improves the storage of information and computational capability while increasing connection delay. Edge computing and fog computing offer similar advantages with decreased latency, but they have restricted storage, capacity, and coverage. Initially, optimization has been employed to overcome the issue of traffic dumping. Conversely, conventional optimization cannot keep up with the tight latency requirements of decision-making in complex systems ranging from milliseconds to sub-seconds. As a result, ML algorithms, particularly reinforcement learning, are gaining popularity since they can swiftly handle offloading issues in dynamic situations involving certain unidentified data. We conduct an analysis of the literature to examine the different techniques utilized to tackle this latency-aware intelligent task offloading issue schemes for cloud, edge, and fog computing. The lessons acquired consequently, from these surveys are then presented in this report. Lastly, we identify some additional avenues for study and problems that must be overcome in order to attain the lowest latency in the task offloading system.

**Keywords:** task offloading, cloud computing, edge computing, fog computing, Internet of things, latency.

**1. Introduction.** The smart IoT is widely consumed, which allows for the connection of numerous electric sensors and gadgets and produces a higher amount of data flow [1]. IoT products that influence and alter our daily activities consist of smart homes, sickness control and avoidance, and connection [2]. These time-critical applications need a greater amount of power, memory and compute resources. While IoT devices are getting stronger all the time, running big programmes on a single device still puts a strain on the battery, Central Processing Unit (CPU), and memory. One solution to the aforementioned problems is computational offloading, in which the computing tasks are moved to another system for execution [3].

The cloud, on the other hand, is a tried-and-true option with established data center infrastructures that might boost the resource capabilities of end devices. Additionally, the cloud has the requisite automation features and instruments to provide end devices with the transparency they want while concealing the complicated technical and logistical problems of this resource augmentation [4]. As a result, shifting computation-heavy activities of resource-heavy apps to centralised Cloud infrastructure is a well-studied approach [4 – 6]. Edge computing is a revolutionary computing platform that places edge servers near clients to perform services including video processing, online gaming, augmented reality (AR), virtual reality (VR), and self-driving automobile applications with recommendation engines. It is required owing to the distance between the final devices and cloud infrastructure, the transport network's reliability, the expense of traveling via the backhaul network, and the wider security surface [7]. Because wireless networks have varied performance, task offloading increases the quantity of data that devices broadcast, implying a longer transmission length and higher transmission energy [8, 9], that results in a shorter device battery life and worse application performance. Edge computing [10] may assist with the previous problem by bringing computing resources (i.e., edge/fog servers1) closer to consumer devices, hence reducing communication distances and, as a result, latency. Edge servers, on the other hand, are significantly fewer than cloud servers, as just a few servers may fit in an edge computing center owing to space constraints and restricted cooling capacity [11]. As an outcome, edge computing will probably provide inadequate processing resources for consumers to finish all activities.

Edge-cloud computing, which mixes the profits of cloud and edge computing, provides one of the approaches that is closest to addressing all of the problems mentioned earlier in order to improve user device battery life and application performance. Because each operation is executed on the consumer device, an edge, or a cloud, edge-cloud computing [12] could offer better computation and transmission performance than edge computing or cloud computing collectively. While task offloading is one of the greatest difficult challenges to solve, it is essential to increase resource utilization efficiency in edge clouds. In an edge-cloud system, task offloading comprises selecting which activities are relocated from customer devices, which edge or cloud the work is allocated to, as well as the server that every offloaded function executes on and in what order. The best offloading solutions must take into account a variety of resources, user requirements, complex networks, user mobility, job dependencies, and other considerations, making these decisions difficult. Compared to fog computing, which occurs within the device's local network, edge computing occurs within the device itself. Since there is no round-trip time to the cloud when using fog nodes, they are especially useful for applications that require low latency. The cloud can be used to run programs that can take a long time to execute as well as to regularly save highlights and information from various fog nodes. Because there are still difficulties for further investigations, we evaluate newly released studies on latency-aware task offloading in cooperative edge, fog, and cloud computing in this work. We begin with a complete description of latency-aware intelligent task offloading in edge-cloud situations, followed by a detailed examination of associated studies efforts based on latency.

Following that, we examine unresolved concerns and propose some exciting areas for further study. We hope that our research on service delivery in edge-cloud computing will be useful to both academics and industry.

The rest of the work is structured as below. The second section offers and thoroughly examines the detailed latency of job offloading in edge-cloud computing. Section 4 compiles the challenges and opportunities for additional research. Lastly, Section 5 brings the paper to a close.

**2. Task Offloading.** Task offloading is a technique of having user equipment operate some computationally expensive programs while wirelessly transmitting the data processed by these applications to an edge server on the condition of weighing continuous or other indicators. As portrayed in Figure 1, the following instances represent possible assessments that could be made while calculating the offload [13].

*Local Execution:* This means that every phase of the calculation is carried out locally. In general, tasks requiring less computational power are the primary goal of this circumstance.

*Full Offloading:* The base station wirelessly connects, offloads, and sends the entire computation to the edge server to be processed. This procedure is sometimes referred to as the binary uninstallation problem and the entire uninstallation difficulty. This problem shows that the edge service application is not isolated it can execute local computations or outsource them to the edge server.

*Partial offloading:* If a portion of the calculation may be done locally, the balance is directed to the edge server for execution.

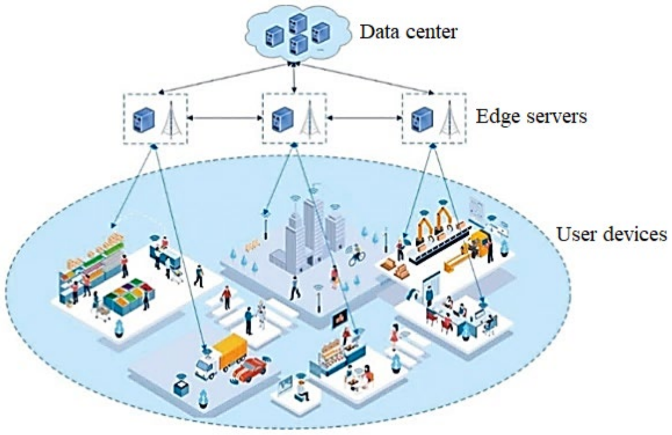

Fig. 1. The architecture of computation offloading 5G networks in edge computing

In Figure 2, a few requirements/important justifications for offloading the tasks are shown. Despite these requirements, the offloading process is also caused by several additional reasons, including energy restrictions, bandwidth restrictions, a lack of server computing power, a lack of available storage, and task size [14].

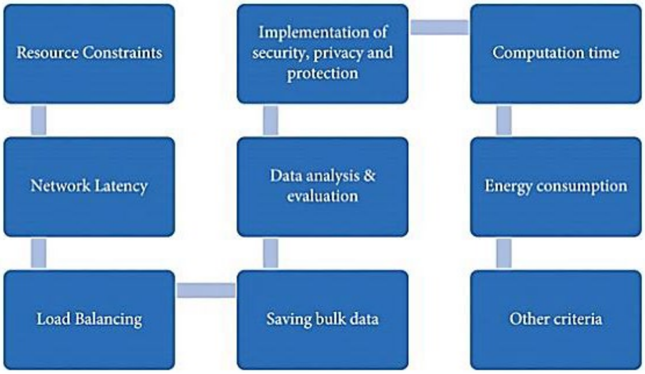

Fig. 2. Task offloading constraints

Some activities have delayed responses, and there is a chance that subsequent jobs might not get resources at all; therefore, latency exists. Therefore, the task offloading (delay) difficulty in cloud-edge-fog computing is explored in their study, while they present advice on how to handle it.

**3. Literature Survey.** The existing research conducted a review on offloading strategy either cloud or edge or fog. This research examined the major and essential difficulty of latency in the task offloading approach used in cloud, edge, fog, edge-cloud, and fog-cloud computing.

**Latency offloading strategy in cloud computing.** Cloud service latency is the amount of duration required for a provider of cloud services to reply to a client request. Latency has a substantial influence on device usage and enjoyment. These concerns might be exacerbated for cloud service communications, which could be especially vulnerable to delay due to several factors.

In [15] the authors provided multilevel full and partial offloading solutions using cloudlet, private, and public cloud servers. According to the simulation findings, the suggested multilayer total and partial offloading systems lower power utilization by roughly 8-9% and 20%, respectively. Here, instead of retrieving the data for processing, the authors were transmitting a brief piece of code. However, the suggested system requires more bandwidth and time-consuming for transferring the little objects. Study [16] provides a simple distributed architecture focused on the alternate direction process of the multipliers procedure to decrease processing latency for all operations. In comparison to current methodologies, the suggested user-assisted computation offloading strategy may minimize calculation latency while retaining high convergence speed. Simulations confirmed the benefits of the suggested user-assisted multi-task offloading approach in terms of minimizing computation latency under diverse conditions. Nonetheless, allocating computing resources will significantly raise the complexity of this challenge.

To improve decision reaction time, in paper [17] the authors proposed a pipelining task offloading method and modeled it as a delay-aware Markov Decision Process (MDP). The researchers then developed a delay-aware MADRL method to lessen the weighted total of job execution latency and power usage. First, the Markov property is rebuilt by enhancing the state space with the most recent state and historical activities. Second, Gate Transformer – extra-long (XL) is introduced to retain the constant input dimension dynamically altered by random transmission delays and to capture the significance of previous events. Third, a sampling approach with a novel loss function with the variations between current with objective state values, as well as the variance between actual state-action value and improved state-action value, are developed to generate state transition trajectories that closely match genuine ones. However, the reaction time difference between the two job-offloading approaches grows as the latency between state collection and action transmission grows.

As a result, in order to meet the aim of minimising average task computation delay while maintaining transmission dependability, a framework for distributed multi-hop computing task offloading depends on an enhanced evolutionary process is suggested by the authors in [18], in which workloads can be recursively distributed among Network Control Protocol (NCPs). The approach optimizes population setup, employs the crossover operator to accelerate convergence, and minimizes the possibility of resource excessive usage resulting from circular schedules by establishing filter chains to filter schedulable nodes before the initialization step. However, finding the best option is tough. In paper [19] the authors introduced a Deep Reinforcement Learning (DRL) depending cooperative task offloading (DCTO) strategy for movement, contact, and load awareness to optimize that. DCTO supports both binary and partial offloading methods, as well as cellular and mm Wave radio access technologies (RATs). DCTO tries to decrease latency by opportunistically switching RATs and offloading mechanisms. When compared to the other evaluated plans, the DRL-based cooperative task offloading proximal policy optimization (DCTO_PPO) method exhibited notable outcomes in terms of reward and TFPS ratio. Consider Table 1.

Table 1. Review on latency offloading strategy in cloud computing

| Ref no. | Algorithm | Performance metric | Simulation tool | Offloading type | Objective | Demerits |
|---|---|---|---|---|---|---|
| [15] | Multilevel full offloading strategy | Power and delay | MATLAB 2015 | Partial and fully offloading | On cloudlet, private, then public cloud servers, multilevel full also partial offloading algorithms are used | Time-consuming process |
| [16] | ADMM algorithm | Latency and improved performance efficiently | MATLAB | Partial offloading | To minimise calculation delay while ensuring user energy availability | The allocation of computing resources will significantly raise the difficulty of this task |
| [17] | Multi-Agent Deep Reinforcement Learning (MADRL) | Decision reaction time and latency | PyTorch 1.10.0 and Python 3.7.11, and the hardware GPU environment is NVIDIA A100-40 GB | Partial offloading | Reduce the total task execution delay and power usage | The time to react variance among the two job offloading strategies grows as the latency between state collection and action transmission grows. This is because the agent of the delay-aware MADRL process doesn't have to wait for the current strategy's response |
| [18] | Improved genetic algorithm | Latency | Python | Partial offloading | To meet the aim of minimising average task computation latency while keeping transmission dependability in mind | The search solution space grows in proportion to the quantity of the task data. Finding the best option is tricky |
| [19] | Deep reinforcement learning (DRL) | Latency | Python 3.7 using PyTorch STable (1.13.1) | Fully and partially offloading | Delay reduction by opportunistic switching of RATs and offloading methods | When the amount of activities is significant the system will not operate well |

**Latency offloading strategy in edge computing.** The authors in [20] suggested a latency-aware computation offloading technique for 5G networks. First, some latency and power usage frameworks for 5G edge computing offloading are described. Then, fine-grained computation offloading is used to lessen total task completion time. The method is lengthy to address the problem of multiuser computation offloading. Extensive simulation tests are carried out in order to validate the efficacy of the suggested tactic. The findings indicate that the suggested offloading approach might efficiently minimise task execution latency. Although restricted resources result in a high rate of resource contention, certain irrational judgments happen owing to a shortage of total network knowledge.

Furthermore, in study [21] the authors offered Nimbus, a task placement and offloading solution for a multi-tier edge-cloud design wherein DL responsibilities for Augmented Reality (AR) applications removed from the pipeline and offloaded to adjacent Graphics Processing Unit (GPU)-powered edge devices. To validate Nimbus' efficiency, the researchers employed trace-driven simulations. The authors demonstrated the potential of Nimbus in enhancing the effectiveness of augmented reality services when offloaded from mobile devices to an edge-cloud structure using a large amount of actual data and experiments. According to the stringent latency limits imposed by immersive applications including AR/VR, the deployment of edge servers in the network is required. As a result, study [22] proposed an innovative design whereby task queue lengths are subject to probabilistic and statistical constraints that are calculated via extreme value theory in order to reduce consumers' electrical usage while trading off the allocated resources for local computation and task offloading. To achieve this purpose, the resources given for local computing and task offloading must be balanced with the energy utilization of the consumers. In this context, a user-server association technique is developed that takes channel quality into consideration as well as server workloads and computing capabilities. By combining matching theory with Lyapunov optimization tools, a two-timescale technique is given in which the long timeframe is utilized to address a user-server association issue, and the short timescale is employed to execute a dynamic job offloading and resource allocation strategy. Currently, ML strategies to reduce latency and dependability are needed.

As a result, study [23] suggested a deep deterministic policy gradient (DDPG) technique for determining offloading strategy. The Deep Deterministic Policy Gradient-based technique, allowing task offloading in multi-server multi-smart mobile devices (SMD) and multi-task mobile edge computing (MEC) with ultra-low latency and expands long-term performance by at least 19%, is detailed. During the resolution process, the dynamic communication environment and the fluctuating server status information across continuous time intervals are taken into account. The researchers can get the offloading object and offloading sections of jobs in every tiny region using the offloading technique. Even so, Offload-MBS and Offload-Local strategies cannot provide a satisfactory user experience in a multi-SMD scenario. In paper [24] the authors presented a two-stage joint optimum offloading approach, optimizing computation along with resource allocation under restricted energy and sensitive latency, to investigate the trade-off between latency and power utilization in low-cost large-scale marine communication. Considering their needs and circumstances, the marine users decide in the first stage whether to offload a computation. In the next phase, the offloading strategy was optimized, taking into consideration the dynamic trade-off between latencies and power consumption, and coordinated with the center cloud servers. Although as the amount of tasks accessed grows, these solutions become increasingly susceptible to delay.

To shorten the overall completion time study [25] demonstrated a simple yet effective offloading strategy for multiuser Edge networks that offloads many important IoT tasks/subtasks to edge servers to minimize anticipated execution time. This is accomplished by explicitly taking into account 1) IoT job topology/schedules; 2) heterogeneous resources on edge servers; and 3) wireless interference in multi-access edge networks. Nonetheless, to increase the offloading efficiency, there is a need to develop a novel optimization algorithm. As a result, study [26] introduced an offloading approach for the joint optimization of telecommunication then processing resources while accounting for the blue trade-off amongst latency and energy utilization. The approach is offered as a tactic for an optimization issue in which the overall power usage is minimized while the execution latency restriction (or deadline) is satisfied. The ideal gearbox power, rate, and proportion of the task to be offloaded are computed analytically and utilized in a solution to satisfy the optimization purpose. Although partial offloading is an extremely complicated procedure that is influenced by a variety of circumstances.

To circumvent the aforementioned constraint, in this study [27] the authors proposed a flexible MEC-based requirement-adaptive partial offloading system to cater to every consumer's unique demands in terms of delay and then consumed energy. The writers gave two normalized variables to account for the dimensional disparities between time and energy and then calculated the computational price of processing operations. To reduce compute overheads while maintaining acceptable delay, task workload, and power restrictions, this paper takes into account realistic fluctuations in user request patterns. However, it increases time complexity. In study [28] the authors concentrated on subtask offloading with logical dependence for IoT applications to reduce weighted task completion latency and power consumption. To investigate subtask dependencies and consider scheduling priorities, subtask-dependent graphs are used specifically. Furthermore, a task offloading technique with dependency assurances for all IoT jobs in multi-server edge networks is provided in order to improve task latency and device power usage.

To minimize total delay, study [29] investigate particularly efficient dynamic spectrum allocation-aided multiuser computation offloading in MEC. To be more exact, the researchers initially concentrated on a static multiuser compute offloading situation before jointly optimizing users' offloading preferences, transmission periods, and resource allocations on Edge Servers (ESs). The authors take into account a dynamic workload and channel offloading scenario for multiuser computation. To efficiently determine the close-to-ideal transmission length in this dynamic context, the researchers offer an online deep reinforcement learning-based technique. However, moving workloads from wireless devices to cloud servers takes time. In paper [30] the authors looked into delay sensitivity-aware computation offloading to maximize task performance benefits by considering into account the state of the network under computation and communication constraints, in addition to the client's tolerance for task delay. The researchers highlight the latency sensitivity of task offloading especially, through an investigation of delay distribution across job groups utilizing a multi-user and multi-server MEC network. The researchers then created a sensitivity-dependent rating system to measure the value of the task's completion and built the Centralised Iterative Redirection Offloading (CIRO) technique to gather all data in the MEC network. By beginning with an early offloading scheme, the CIRO technique allows IoT gadgets to interact along with iteratively redirecting task offloading decisions to boost the offloading scheme until it converges. The summary is presented in Table 2.

Table 2. Review on latency offloading strategy in edge computing

| Ref no. | Algorithm | Performance metric | Simulation tool | System | Offloading type | Objective | Demerits |
|---|---|---|---|---|---|---|---|
| [20] | Delay-aware optimization algorithm | Latency and energy consumption | - | Multi-user | Fully offloading | Decomposing a computer task into numerous subtasks can decrease task delay & total completion time | Although restricted resources result in an elevated amount of resource contention, certain irrational judgments happen owing to a shortage of general network knowledge |
| [21] | Nimbus task offloading approach | Network latency, inference, and queuing time | NVIDIA Triton | Multi-user | Fully offloading | End-user delay & energy expenses on portable devices should be kept to a minimum | According to the stringent latency limits imposed by immersive systems including AR/VR, the installation of edge servers in the network is required |
| [22] | Ultra reliable and low latency communi-cation (URLLC) | Task computation and delay performance | - | Multi-user | Partially offloading and fully offloading | To reduce users' energy usage while allocating resources for local computing & task offloading | The researchers will look at distributed ML approaches to minimize latency as well as dependability even more |
| [23] | Deep deterministic policy gradient approach | latency, effective server utilization, and frequent mobility | The authors consider a scenario that, a network topology of 10 km × 10 km consists of one Macro Base Station (MBS), 100 Small Base stations (SBSs), and trajectories of SMDs. SBSs are uniformly deployed in the MBS signal coverage area located at the center of the MEC network | Multi-user | Parallel offloading | To consume the Markov decision strategy to reduce the time required to complete all tasks & turn this issue into a deep reinforcement learning-based offloading strategy | In a multi-SMD scenario, the Offload-MBS and Offload-Local strategies are unable to offer a satisfactory user interface |
| [24] | Joint Optimal Offloading Algorithm for Ship Users in Mobile Edge Computing (JOOA) | Resource limitation and latency | The channel bandwidth is 4MHZ, there are 10 channels, and the bandwidth of each channel is nel is –60dBm | Multi-user | - | The goal of this study is to look at the trade-off between delay and power usage in cheap-cost large-scale marine communi-cation | Although as the amount of jobs accessed grows, these solutions become increasingly vulnerable to low latency |

*Continuation of Table 2*

| Ref no. | Algorithm | Performance metric | Simulation tool | System | Offloading type | Objective | Demerits |
|---|---|---|---|---|---|---|---|
| [25] | EFO (earliest finish-time offloading) Algorithm | Overall completion time and delay | UDN network | Multi-user | Binary offloading | To reduce the overall completion time | To optimize the offloading efficiency machine-learning-based approaches need to develop |
| [26] | Joint optimization of the communi-cation and computational resources | latency constraint, task complexity | Mobile edge computing | Multi-user | Full and partial offloading | Joint optimization for transmission and computational resources by taking the blue trade-off between usage of energy and delay into account | Although partial offloading is an extremely complicated procedure that is influenced by a variety of circumstances |
| [27] | Delay-Energy Balanced Partial Offloading (DEB-PO) scheme | Energy consumption and delay | MATLAB using CVX on a desktop computer with an Intel Core i7-4790 3.60GHz CPU and 16GB RAM | Multi-user | Partial offloading | To reduce computing overheads by optimizing offloading workloads with power together over tolerable latency restrictions | It increases time complexity |
| [28] | SDG-task offloading algorithm | latency and energy consumption | Multi-server edge networks | Multi-user | Partial offloading | To reduce the number of weighted task execution delays & power usage | SDG tasks have less parallelism than parallel structure tasks. In contrast, because of logical interdependence among tasks, subtasks run on various servers must communicate data, increasing communication costs |
| [29] | Deep reinforcement learning-based online algorithm | Latency | MATLAB R2018a with Intel Core i5 2.4 GHz CPU and 8G RAM | Multi-user | Partial and fully offloading | In MEC, optimum dynamic spectrum allocation-assisted multiuser offloading of computation is used to reduce total latency | Workload migration from wireless devices to cloud servers takes time |
| [30] | Centralized Iterative Redirection Offloading (CIRO) algorithm | Task execution delay | Python | Multi-user and multi-server | Partial offloading | To optimize the computation offloading strategy to maximize the task execution utility | Complexity is increased |

The aforementioned studies have contributed major improvements to computation offloading in MEC. Based on the preceding study, we explore the partial offloading mode and resource allocation in a system with several users in the present study. Although balancing edge servers for efficient server utilization hasn't been fully researched in either binary or partial offloading modes, the frequent mobility of several clients in a multi-user MEC system must be considered.

**Latency offloading strategy in edge-cloud computing.** Cloud computing is considered as a realistic and desirable technique to assist this development due to its ability to provide on-demand access to a massive pool of computational power for service operations and data analytics [31, 32].

To state the limitations of cloud computing, the edge computing concept has emerged, resulting in the creation of a pool of virtually operated storage and processing assets at the network's edge that are within a variety of IoT devices and therefore minimize latency gaps [33, 34]. In addition, the edge computing design prohibits outsourcing task pre-processing and minimizes massive data uploading and downloading, both of these reduce overall service time [35, 36].

However, the effective administration of edge-cloud computing capabilities along with the managing of computational tasks for applications that are latency-sensitive is an essential problem [37], which could require the proposal of a successful task scheduling approach to enhance total service performance while decreasing offloaded task delay.

In relation to the preceding justification, it ought to be observed that the feasible approach to scheduling tasks that are offloaded on the edge-cloud framework requires into account different application features (computational, communications, and latency), resource heterogeneity, and resource utilization, that could be shown as a dynamic multi-objective optimization issue that changes as time goes on [38].

Study [39] suggested a unique methodology for offloading duties in an edge-cloud network to reduce total service time for latency-sensitive applications. This technique employs fuzzy logic techniques and considers resource utilization along with heterogeneity, in addition to application features (including CPU demand, network demand, and delay sensitivity). Additionally, other factors like network congestion have an impact on the network time of transmitted data. As an outcome, approaches like reinforcement learning may prove useful for teaching the framework to make correct judgments but also for monitoring the effectiveness of offloading decisions by observing every action. Furthermore, because this operation trades utilization for a shorter total service duration, it may squander resources at the edge, particularly processing power. More energy-efficiency solutions (including Virtual Machine migration and auto-scaling horizontal/vertical) need to be added in the forthcoming to address this problem and strike stability amid serving the demands of applications and then effectively applying the Edge/Cloud computing resources (Figure 3).

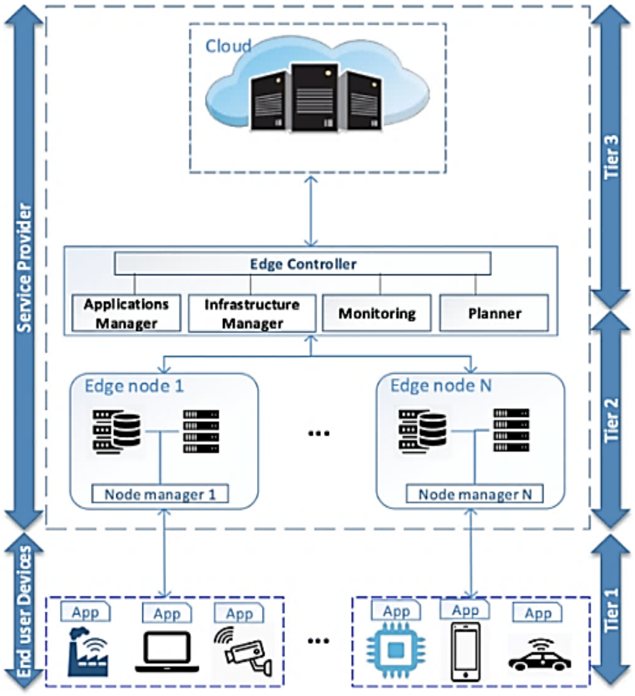

Fig. 3. An overview of the Edge-Cloud system

In a multi-user situation, in paper [40] the authors suggested a multi-tier edge-cloud computing method for work offloading that is delay-optimal. The difficulty is given as an optimization framework consuming Integer Linear Programming (ILP) approaches to decrease the entire service time of Unmanned Aerial Vehicles. Furthermore, the researchers organized the task offloading challenge as an Integer Linear Programming (ILP) model in order to shorten the overall service time of UAVs. A task offloading algorithm that is more effective was also created using the ILP model's insights. Long latency is, nevertheless, essential for cloud computation. As a result, study [41] proposed a back-pressure algorithm (BMDCO) that created a computational offloading technique for reducing latency to acquire the offloading choice and the number of jobs that may be offloaded in order to minimize computational delay while improving reaction rate. The researchers then designed an offloading approach to lower the queue length of jobs in every time slot by decreasing the Lyapunov drift optimization issue in order to achieve queue stability while enhancing offloading performance. In study [41] the authors do not include consuming power as one of the tasks in an edge-cloud system (Table 3).

Table 3. Review on latency offloading strategy in edge-cloud computing

| Ref no. | Algorithm | Performance metric | Simulation tool | System | Offloading type | Objective | Demerits |
|---|---|---|---|---|---|---|---|
| [39] | Fuzzy logic algorithm | Delay, time, and resource utilization | Edge-Cloud environment | Multi-user | Partial offloading | To meet the needs of latency-sensitive IoT programs while effectively utilizing resources in Edge-Cloud scenarios | Job duration, the quantity of transmitted data for uploading and downloading data, that may not always be precise owing to the influence of another element |
| [40] | Delay-optimal task offloading algorithm | Latency and safety concerns are taken into account | MATLAB | Multi-user | Partial offloading | UAVs' entire service duration should be kept as short as possible | Cloud execution necessitates a high level of delay |
| [41] | BMDCO algorithm | Delay and Response rate | MATLAB R2016b, 8 GB RAM, Intel i5 3.20 GHz CPU, and Windows 10 | - | - | To shorten the calculation time and increase the response rate | The energy consumption of the task is considered |
| [42] | Resource scheduling policy and routing algorithm | Latency and better energy efficiency | Edge-Cloud environment | Multi-user | Partial offloading | To decrease the general system latency of all mobile phones | In our computation model, computing at local devices is not considered |
| [43] | A joint task offloading and scheduling (JTOS) framework | Delay and energy | SDN Fog Cloud Simulation Tool | Multi-user | Full offloading | To minimize the hybrid delay of all applications | For IoT services, the JTOS fails to offer mobility-aware or location-aware services. This effort is still hampered by safety holes in the fog cloud network |

To reduce energy with latency, paper [42] examined the interface of cloud and edge computing, wherein mobile device operations might be performed a portion at the edge node with the cloud server. To begin, the weighted-sum latency of all mobile gadgets is decreased by developing a combined interaction and compute resource allocation issue. The closed-form optimum task-splitting approach is then resultant using the normalized backhaul connection capacity and the normalized cloud compute capacity. Although the computational paradigm, computing at local devices, is not taken into account, necessitating the use of linear programming to distribute task offloading and scheduling. Then, the authors [43] employ combinatorial integer linear programming (CILP) to solve joint scheduling and management issues. The researchers suggested a joint task offloading and scheduling (JTOS) structure in response to the problem. JTOS has task offloading, sequencing, scheduling, searching, and breakdown parts. Hence, Figure 3 shows the overview of the Edge-Cloud system.

Edge clouds will help meet the strict latency demands of the future types of applications that run in real-time including AR and VR by moving computing, memory, and networking resources nearer to consumer devices.

**Latency offloading strategy in fog computing.** Fog computing is a distributed computing methodology that is quickly gaining traction because it delivers cloud-like services near to end devices. It improves the computational abilities of mobile nodes and IoT devices by offering cloud-like computing and storage functionalities with lower latency and lower bandwidth usage.

The issue of offloading for latency-sensitive applications may have a solution due to fog computing. Fog devices will significantly minimize processing latency if they are used to process the tasks. This also helps to reduce network congestion that can occur when multiple IoT devices submit information and activities to cloud servers at the same time.

In study [44] the authors developed a dynamic service that chooses jobs that can sustain another offloading. The offloading destination node is selected depending on job type, latency constraints, and quantity of resources required. Three heuristic offloading approaches are proposed in this paper. Every method focuses on a specific job type. An overloaded fog node is allowed to send out one offloading request to implement any of these techniques based on the task offloading priority. Requests for offloading are made to a Software Defined Networking (SDN) controller. The fog node and controller decide the amount of jobs that are offloaded. Although the impact of movement on the intended service wasn't taken into account. To minimize the execution cost of a task burden offloaded to fog devices, study [45] proposed a Quality of Service (QoS)-aware task offloading solution using a brand-new, nature-inspired optimization technique termed the Smart Flower Optimization algorithm (SFOA). When choosing the best fog nodes to offload computing work to, the suggested technique takes into account QoS characteristics including task deadlines and financial limitations.

Study [46] offered a cost-cutting offloading approach in fog computing, which is a weighted average of energy use and general processing delay for every end-user operation. The researchers leveraged the heterogeneous nature of fog computing nodes with variable CPU frequencies to handle jobs. The individual computational capacity of the fog node was found to be a significant factor in the local task processing time, which in turn affected the overall task processing cost. The authors used semidefinite relaxation (SDR) to an optimization problem to resolve the non-convex optimization problem. The offloading profile of the fog node, in particular, gives insight into work offloading for every fog node with regard to energy and processing time.

The dynamic collaborative task offloading (DCTO) technique, introduced by the authors in [47], depends on the resource circumstances of fog devices. To minimize task execution latency, a job might be performed by a single fog device or by numerous fog devices via parallel processing of subtasks. By comparing the suggested offloading approach to the current methods, rough simulation outcomes indicated significant benefits in drastically decreasing the average latency in networks with an extensive amount of service requests and a diversified fog climate. Certain fogs that have restricted resources are incapable of handling all of the data from a single task. As a result, in paper [48] the authors developed Fog Resource aware Adaptive Task Offloading (FRATO), an architecture for IoT-cloud systems that employs an adaptive task offloading strategy to enable the fastest service providing feasible. FRATO focuses primarily on the usage of the fog resource to establish an optimum offloading technique, involving a collaborative task offloading technique that utilizes the notion of data fragments. To efficiently deploy optimized offloading technologies in resource-constrained environments, two distributed fog resource allocation techniques, task priority-based resource allocation (TPRA) and maximum resource utilization-based allocation (MaxRU), are developed. However, in the iterative procedure, it has a poor convergence rate.

Paper [49] presented the meta-heuristic scheduler Smart Ant Colony Optimisation (SACO) task offloading approach to offload jobs for IoT-sensor applications in a fog environment in order to minimize compute and transmission delay. The numerical results of the proposed Smart Ant Colony Optimization (SACO) method indicate the task offloading algorithm's considerable decrease in latency. In comparison to the Round Robin (RR), throttled procedure, and two bio-inspired methods modified PSO and Bee life algorithm (MPSO and BLA), computational findings show that the suggested

SACO technique has significantly lower latency in task offloading for IoT-sensor functions. Study [50] suggested a Globally Optimal Multi-objective Optimisation method for Task Offloading (GOMOTO) that utilized the performance model to successfully and swiftly realize task offloading. The performance framework for task offloading in a fog computing situation is built utilising network calculus theory. The outcomes display that their suggested approach and technique may successfully reduce the system's total latency and power usage while improving network Quality of Service (QoS). As a result, achieving prompt and dependable job offloading in the fog layer is a significant problem for fog computing technology.

In [51] the authors proposed the distributed computation offloading architecture (DISCO) for offloading splittable workloads in order to obtain a decreased execution period for the fog computing scenario. Through an extensive simulation study, the presented approaches reveal possible benefits in decreasing average latency. The model given in this work can suggest some directions for further research. First, a Helper Node (HN) that can process several tasks or subtasks can be employed with a many-to-many matching model. Several HN users can work together to manage a group of jobs or subtasks with the aim of optimizing the method's latency performance. Although the scheduling of tasks or subtasks for every HN is considered in this circumstance, the externality is also considered (Table 4).

Table 4. Review on latency offloading strategy in fog computing

| Ref no. | Algorithm | Performance metric | Simulation tool | System | Offloading type | Objective | Demerits |
|---|---|---|---|---|---|---|---|
| [44] | Latency-based Fog-to-Fog Offloading LFFO algorithm | Latency | Apache NetBeans. Algorithms were written in Java. 16 GB of RAM | Multi-user | Fully offloading | To shorten the calculation time as well as increase the response rate | The effect of mobility on the proposed service was not considered |
| [45] | Smart Flower Optimization Algorithm (SFOA) | Delay | Workload-Fog scenarios | Multi-user | - | The subject under consideration is given as the problem of minimizing the cost of implementation of a work burden offloaded to fog devices | Time-consuming process |
| [46] | Proposed Offloading Strategy | Minimization of the total system cost and task processing delay | MATLAB | Multi-user | Fully offloading | Under energy along with latency limitations, determine the appropriate quantity of task data to be handled locally or offloaded to the preferred fog node and remote cloud | If the primary fog node's computational resource is insufficient, the primary fog node chooses to offload the work to a neighbouring fog node and cloud |

*Continuation of Table 4*

| Ref no. | Algorithm | Performance metric | Simulation tool | System | Offloading type | Objective | Demerits |
|---|---|---|---|---|---|---|---|
| [47] | A dynamic collaborative task offloading (DCTO) approach | Reduce the task execution delay | Python | Multi-user | Full and partial offloading | To meet the delay reduction goal | Heavy workloads (i.e., a = 10 MB) have considerable effects on the fog computing environment because certain fogs with few resources aren't able to handle the entire data of just one task |
| [48] | FRATO (Fog Resource aware Adaptive Task Offloading) – a framework | Delay | Powerful fogs have 1 GB of RAM and 6 GB of storage | Multi-user | Full task offloading | To deliver context-aware IoT computing capabilities with minimal latency | The PSO approach has the disadvantage of being easier to slide into a local optimum in high-dimensional space while maintaining a low rate of convergence in the repeating phase |
| [49] | Smart Ant Colony Optimization (SACO) | Latency and processing time | 2018b MATLAB. It was run on a PC with a 2,80 GHz Intel(R) Core(TM) i7 CPU M640@2.80 GHz and 8 GB RAM | - | - | Minimizing task offloading time in IoT-Fog environment by considering computation and communication latency | Power consumption is not considered |
| [50] | Globally Optimal Multi-objective Optimization algorithm for Task Offloading (GOMOTO) | Latency and Energy consumption | JDK 12 | Multi-user | - | The complete FN performance concept is defined as the linear weighted total of equivalent delay, power usage, and storage space | As a result, providing consistent and reliable work offloading in the fog layer is a big challenge for fog computing design |
| [51] | Distributed computation offloading framework (DISCO) | To reduce the average delay | SimPy library in Python | - | Partial offloading | To decrease the processing time in the fog computing atmosphere | Computational complexity |

Despite technical breakthroughs, IoT and smartphones remain resource-restricted. Because of their restricted memory, processing, and networking abilities, these gadgets might be unable to independently analyze massive quantities of information or perform compute-intensive processes over an assured time period. If the bulk of data or the complexity of apps grows, these electronics will be impossible to run them for a reasonable length of period.

**Latency offloading strategy in cloud-fog computing.** The IoT layer (Tier 1), Fog layer (Tier 2), and Cloud layer (Tier 3) are three tiers of a three-layered hierarchical structure Figure 4 that are taken into consideration.

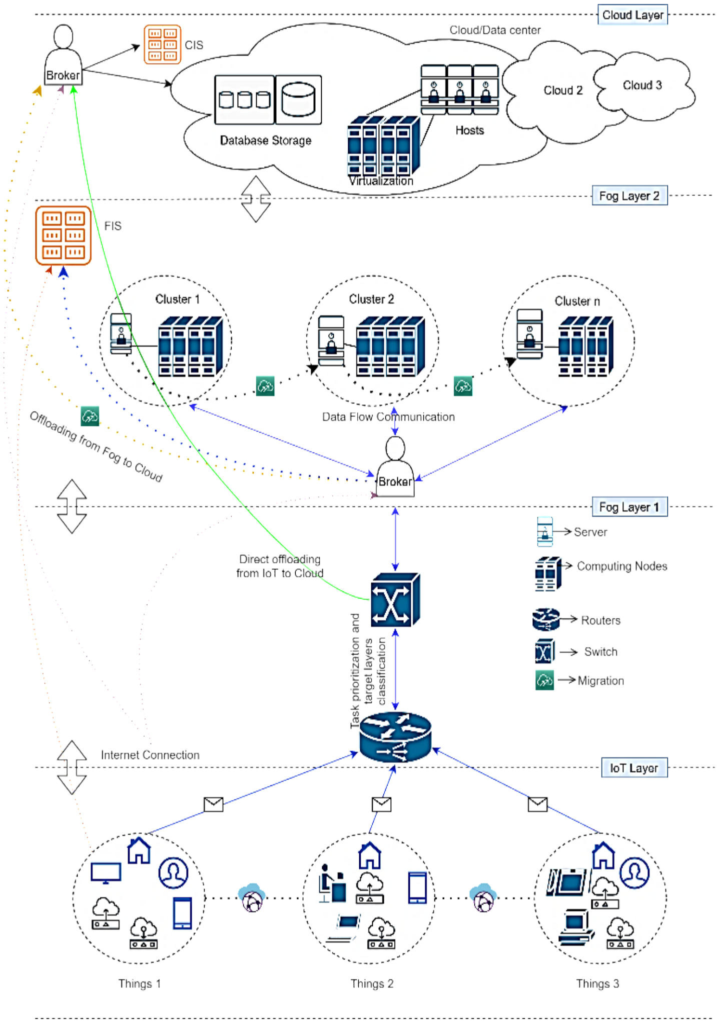

Fig. 4. A three-layered Framework for IoT-Fog-Cloud [52]

In paper [52] the authors suggested a fuzzy logic approach for task prioritization that uses resource needs and deadlines in order to benefit from the effectiveness of the underlying computing nodes for jobs' heterogeneity and computational requirements with deadline constraints. For effective planning, an elitism-based multi-population Jaya is suggested to map these distinct groupings of activities to a cluster aggregation of computationally capable heterogeneous computing nodes.

A compatibility-based heuristic offloading approach is created as well to locate compatible computing nodes to offload computations while considering the availability of resources and delay in communication from relevant IoT gadgets into account. These activities necessitate the use of increasing resource – and memory-intensive computer nodes for execution. In study [53] the authors suggested a multi-armed bandit (MAB) theory-based optimized offloading system to reduce mean computing task offloading latency. USV cluster nodes may employ this strategy to minimize the average computation task offloading latency by recognizing the potential computing capabilities of their neighboring team nodes. The Adaptive Upper Confidence Boundary (AUCB) method is an optimized technique with related simulations to evaluate performance. Further complicated tasks were not met by this technique.

Less bandwidth equals less delay, which enhances reaction time. In paper [54] the authors propose fog computing as middleware since it delivers operations to the network's edge effectively and serves delay-sensitive activities. Since creating a task-offloading strategy is complex, this research paper tackles the problem that proposes an intelligent task-offloading framework. They cannot operate for an extended amount of time or could fail to make helpful judgments in complicated scenarios, and the main goal of the task offloading problem is to minimize latency and power usage (Table 5).

Study [55] examined the topic of task offloading delay decrease in a hierarchical fog computing Cloud Radio Access Network (C-RAN) with three tiers of computational services: MEC servers in radio units, MEC servers in distant units, and cloud-based services in centralized units. The receive beamforming vectors, job distribution, processing performance for offloaded tasks on every server, and transmission bandwidth split of front haul links are all optimized by answering the given mixed integer programming issue.

To offer a realistic setting for the evaluation of the suggested task-loading strategies, in study [56] the authors compile a set of Python tasks. In addition, a four-tier structure is developed in this research to identify the appropriate decision-maker for the job at hand. The examiners then classify the issue as a population (evolutionary) game that employs Maynard replicator dynamics. The primary optimization goals in this study are to reduce time and energy consumption. Finally, the authors utilized MATLAB to simulate the suggested strategy, paying particular attention to the usage of accurate values and parameters. As a result, the NP-hard optimization on the character of the issue has restricted computing resources. In paper [57] the authors presented fog computing as an extra computing model for cloud computing to reduce latency. IoT-based systems run with greater effectiveness when combined with fog computing. When fog is unable to perform an operation owing to insufficient capacity, heavy calculations are offloaded to the cloud. However, understanding when to go from fog to cloud is crucial. The decision to move work from fog to cloud needs to be taken cautiously, and this research provides a solution.

Table 5. Latency offloading strategy in cloud-fog computing

| Ref no. | Algorithm | Performance metric | Simulation tool | System | Offloading type | Objective | Demerits |
|---|---|---|---|---|---|---|---|
| [52] | Elitism-based multi-population Jaya algorithm | Minimise offloading time for latency-sensitive applications while keeping deadline and latency constraints in mind | iFogSim | Multi-user | Binary offloading | To provide a latency-aware offloading technique that takes into account task priorities whilst fulfilling deadlines along with competing QoS restrictions | These activities necessitate the use of increasing resource- and memory-intensive computer nodes for execution |
| [53] | Adaptive Upper Confidence Boundary (AUCB) algorithm | delay | iFogSim | Multi-user | Fully offloading | To minimize average computation task offloading delay | This algorithm did not satisfy more complex tasks |
| [54] | Intelligent task offloading model | Bandwidth delay and response time | SFogSim | Multi-user | Multi fog offloading | Less bandwidth equals less delay, which increases reaction time | They cannot function for an extended amount of time and might be unable to arrive at effective conclusions in complex scenarios |
| [55] | C-RAN network | Delay | Fog-computing network | Multi-user | Binary offloading | To improve the delay performance of task offloading | limited computational capabilities |
| [56] | Polynomial Parity Arguments on Directed Graphs (PPAD) algorithm | Delay and energy | Python and Matlab` | Multi-user | Binary offloading | The basic goals of the task offloading issue are to minimize delay and energy expenditure | The NP-hard character of this issue, as well as the presence of several unfounded presumptions in the offered answers, represents an essential study need in this domain |
| [57] | MTFCT algorithm | System efficiency and delay | FogSim | Multi-user | Partial offloading | To minimize the delay | To allow task migrations from node to node in the fog computing circumstances, as well as an improved strategy to manage the job on the cloud data center, either work is offloaded from a portable device or any fog node |
| [58] | Branch-and-bound for MILP | Energy and delay | Matlab2018a | Multi-user | Partial and full offloading | Minimizing the total energy consumption and latency of communication | Mobility of different network components is not considered |

In [58] by decreasing the overall power usage of IoT users about communication and computation latency, the authors are putting forth an ideal offloading choice technique and resource allocation scheme in a fog atmosphere. To optimally offload jobs to the fog node, an optimization issue is constructed, with an upgraded branch-and-bound tree proposed for issue assessment. Among the characteristics used by the researchers to assess performance include the average task delay, average power usage per job, and median remaining power per IoT node.

Cloud platforms have already penetrated all sectors of business and left an impression on the economic landscape. Each enterprise is searching for cloud-based business solutions because of the advantages that have a significant impact on a corporation's operational costs. Fog Computing is helping the Cloud Computing platform with increasing its market share and providing technologies depending on the Cloud with a new lease on life.

**Latency offloading strategy in IoT-edge-cloud.** In paper [59] the authors investigated the workload distribution issue in an IoT edge-cloud computing structure with guaranteed delays and energy efficiency. The researchers created a latency-based task allocation problem that provides the optimum workload distributions over local edge servers, neighbouring edge servers, and the cloud while ensuring minimal power usage and delay. The difficulty is then solved by consuming a delay-based workload allocation (DBWA) method that utilises the Lyapunov drift-plus-penalty theorem. Meanwhile, dynamic traffic patterns of edge and cloud servers, in addition to their diverse computing capacities provide an issue to task allocation. Paper [60] described a block-chain situation whereby secured task offloading may done by collaborating with edge computing and cloud computing to reduce the usage of energy of the IoT device with delay limitations. In addition, the writers proposed an IoT-Edge-Cloud computing framework for block-chain that leverages both Mobile Cloud Computing (MCC) and MEC, with MEC servers given decreased latency computing services and MCC servers providing more powerful processing capacity. Moreover, they provide an energy-efficient dynamic task offloading (EEDTO) approach by determining the appropriate processing location online, whether on an IoT device, an MEC server, or an MCC server, in order to minimize both power consumption and task response time. The Lyapunov optimization approach is employed to regulate compute and communiqué expenditures for diverse applications, including dynamic alterations to wireless settings. Offloading duties from IoT gadgets to MEC/MCC servers may not be advantageous, particularly if network bandwidth is limited (Table 6).

Table 6. Latency offloading strategy in IoT-edge-cloud

| Ref no. | Algorithm | Performance metric | Simulation tool | System | Offloading type | Objective | Demerits |
|---|---|---|---|---|---|---|---|
| [59] | Delay-base workload allocation (DBWA) algorithm | Transmission power use and network transmission delay | Matlab and C++ | - | - | To minimize the drift-plus-penalty, as well as the method's power usage, while providing a per-job granular delay guarantee | The dynamic traffic patterns of edge and cloud servers, along with their diverse computing abilities pose an issue to task allocation |
| [60] | Energy-efficient dynamic task offloading (EEDTO) algorithm | Delay and energy consumption | Mobile blockchain network | Single user | Partial offloading | With time constraints, the objective is to lower the electrical consumption of the IoT device | Offloading duties from IoT gadgets to MEC/MCC processors might not prove advantageous, particularly if network bandwidth is limited |
| [61] | Multi-hop cooperative-communi-cation model (MCCM) | QoS, and delay | Matlab | - | - | Aiming for QoS improvements | However, due to resource constraints in the edge server system, an edge-based IIoT may be unable to support resource-intensive activities |
| [62] | Deep Meta Reinfor-cement Learning-based Offloading (DMRO) | Delay, minimise computing strain, and enhance job processing efficiency | - | Multi-user | Full offloading | To identify the best offloading system in terms of weighted delay and power usage | When the MEC surroundings shift, the algorithm quickly converges, so only a few learning steps are required to provide low-cost offloading alternatives |
| [63] | DDMTO (Distributed Deep Meta learning-driven Task Offloading) | CPU energies, executions, network demands, delays, as well as resource utilization, heterogeneities, and overall times are all considered | Python | Multi-user | Partial offloading | To enhance service time and resource consumption | Offloading tasks to the edge/cloud servers might not be ideal, particularly if network bandwidth is constrained |

Study [61] investigated the multi-hop computation-offloading issue for the IIoT-edge-cloud computing paradigm by employing a game-theoretical method for distributed computation offloading that is QoS aware. To lower the computational period and power usage of every task, the writers initially investigated the computation-offloading and communication-routing difficulties. The overall challenge is then formulated as a possible game whereby the IIoT devices select their computation-offloading approaches. Secondly, a free-bound technique with a finite enhancement route to a Nash equilibrium is adopted. Finally, the writers created two distributed methods capable of reaching the Nash equilibrium that suggest a multi-hop cooperative messaging network. Meanwhile, due to resource constraints in the edge server, an edge-based IIoT may be unable to support resource-intensive activities.

Paper [62] suggested a Deep Meta Reinforcement Learning-based Offloading (DMRO) method that employs numerous concurrent DNNs along with Q-learning to generate precise offloading judgments in direction to recognize the optimal offloading system that lessens the weighted delay along with power use. The optimum offloading method from the IoT environment may be acquired quickly and flexibly by integrating DL's perceptual capacity, reinforcement learning's decision-making strength, and meta-learning's rapid environment learning ability. After that, to resolve the problems of poor portability and ensure that DNNs are employed for creating successful and productive offloading decisions, study [63] offer DDMTO (Distributed Deep Meta learning-driven Task Offloading). The BP method computes the outputs of these networks utilizing inputs from hidden layers. Based on discrepancies, errors are tracked from intended outputs to hidden layers and from hidden layers to input layers. Neuron weights alter when the flow returns. Epochs are cycles that travel from inputs to outputs and back.

**3. Problem Statement.** The researchers mentioned here have provided major improvements to task offloading in cloud, edge, and fog computing.

– If there is an excessive amount of task-offloading consumers, certain offloaded processes require waiting for sufficient processing capacity, incurring a delay. The waiting time for task processing at the server can't be overlooked in this scenario and must be considered.

– Task offloading in multi-user systems in MEC ought to be investigated, since balancing edge servers to optimal server utilization hasn't been well investigated in either binary or partial offloading modes.

– Moreover, there is a delay in resource provisioning for an edge or cloud, such as the seconds or minutes it takes a cloud to activate a virtual machine instance. Hence, this delay in task offloading could lead to violations of QoS standards, such as response times.

– Furthermore, edge-cloud computing necessitates the assessment of provisioning delay, as well as task offloading which is aware of the resource provisioning delay. The calculation of provisioning delay is complex since there are numerous aspects to consider, including the heterogeneity of the resource collection, the software required for offloaded activities, network configuration, time of day, location of edge or cloud, and furthermore.

– To overcome the above-mentioned consequences we provided a suggestion which is discussed in the following section.

**4. Objectives.**

– In cloud computing, offloading strategies can accelerate the computation of certain parallel ML methods by leveraging their high degree of parallelism, hence reducing latency.

– Furthermore, DL algorithms are extremely effective in addressing the inherent difficulties of edge-cloud computing platforms, where neural network algorithms may decrease multiplication processes during model solutions, also accelerate the method of convergence.

– Various optimization algorithms are introduced to minimize latency and convergence time, hence expediting offloading choices. In this manner, we plan to assess the effectiveness of procedures when offering time-sensitive operations with preset deadlines.

– As a result, we want to create a unique latency-aware intelligent task offloading strategy for edge or cloud computing as a component of our next project.

**5. Future Directions.** This study proposes some open directions for addressing the latency restrictions in cloud-edge-fog computing.

– As discussed in the overview, the division of data is a critical technique for decreasing latency in complex heterogeneous surroundings. Anyway, because of the variety of input data structures in actual applications, various division techniques are required to improve or optimize system efficiency. Furthermore, the information might be specifically split into distinct properties including size. The optimization may therefore be framed to determine the ideal amount of data subsets and related data subset sizes for optimizing system efficiency.

– A novel swarm-based optimization technique is needed to consider the average metrics such as queue length and latency.

– In addition, traditional as well as classic logic is an intelligent system suitable for being used in those revolutionized sectors where decision-making help is required; consequently, it may be an effective approach to keep up the existing research conducted with the fog-cloud job offloading technique.

– Work to come may collaborate on local, edge, and cloud processing resources to improve performance even more.

**6. Conclusion.** A complete study of the system's efficiency with regard to latency in cloud, edge, and fog computing is one of the prospective research. In this research, we present a strategy for detecting associated research in edge-cloud-fog computing based on task type, offloading methodology, and purpose. After that, we thoroughly examine previously published studies that are related to task offloading. Due to investigations on latency-aware task offloading in edge-cloud computing, several challenges must be investigated before edge-cloud computation may be utilized for providing services. As a consequence, we discussed a few of these difficulties, in addition to possible additional research on task offloading latency in edge-cloud-fog computing. For academics, and industry groups interested in edge clouds, we trust that our survey work is beneficial.

## References

1. Wang F., Zhu M., Wang M., Khosravi M.R., Ni Q., Yu S., Qi L. 6G-enabled short-term forecasting for large-scale traffic flow in massive IoT based on time-aware locality-sensitive hashing. IEEE Internet of Things Journal. 2020. vol. 8. no. 7. pp. 5321–5331.
2. Wei D., Ning H., Shi F., Wan Y., Xu J., Yang S., Zhu L. Dataflow management in the internet of things: Sensing, control, and security. Tsinghua Science and Technology. 2021. vol. 26. no. 6. pp. 918–930.
3. Zheng T., Wan J., Zhang J., Jiang C., Jia G. A survey of computation offloading in edge computing. In 2020 International Conference on Computer, Information and Telecommunication Systems (CITS). IEEE, 2020. pp. 1–6.
4. Saeik F., Avgeris M., Spatharakis D., Santi N., Dechouniotis D., Violos J., Papavassiliou S. Task offloading in Edge and Cloud Computing: A survey on mathematical, artificial intelligence and control theory solutions. Computer Networks. 2021. vol. 195. no.108177.
5. Zhao T., Zhou S., Guo X., Zhao Y., Niu Z. A cooperative scheduling scheme of local cloud and internet cloud for delay-aware mobile cloud computing. IEEE globecom workshops (GC Wkshps). IEEE, 2015. pp. 1–6.
6. Xu F., Yang W., Li H. Computation offloading algorithm for cloud robot based on improved game theory. Computers & Electrical Engineering. 2020. vol. 87. no. 106764.
7. Shakarami A., Ghobaei-Arani M., Masdari M., Hosseinzadeh M. A survey on the computation offloading approaches in mobile edge/cloud computing environment: a stochastic-based perspective. Journal of Grid Computing. 2020. vol. 18. pp. 639–671.
8. Guo S., Zeng D., Gu L., Luo J. When green energy meets cloud radio access network: Joint optimization towards brown energy minimization. Mobile Networks and Applications. 2019. vol. 24. pp. 962–970.

9. Dai H.N., Wong R.C.W., Wang H., Zheng Z., Vasilakos A.V. Big data analytics for large-scale wireless networks: Challenges and opportunities. ACM Computing Surveys (CSUR). 2019. vol. 52. no. 5. pp. 1–36.

10. Hong C.H., Varghese, B. Resource management in fog/edge computing: a survey on architectures, infrastructure, and algorithms. ACM Computing Surveys (CSUR). 2019. vol. 52. no. 5. pp. 1–36.

11. Xu Z., Liang W., Jia M., Huang M., Mao G. Task offloading with network function requirements in a mobile edge-cloud network. IEEE Transactions on Mobile Computing. 2018. vol. 18. no. 11. pp. 2672–2685.

12. Ren J., Zhang D., He S., Zhang Y., Li T. A survey on end-edge-cloud orchestrated network computing paradigms: Transparent computing, mobile edge computing, fog computing, and cloudlet. ACM Computing Surveys (CSUR). 2019. vol. 52. no. 6. pp. 1–36.

13. Zhang Z., Li C., Peng S., Pei X. A new task offloading algorithm in edge computing. EURASIP Journal on Wireless Communications and Networking. 2021. vol. 2021. pp. 1–21.

14. You C., Huang K., Chae H., Kim B.H. Energy-efficient resource allocation for mobile-edge computation offloading. IEEE Transactions on Wireless Communications. 2016. vol. 16. no. 3. pp. 1397–1411.

15. De D., Mukherjee A., Guha Roy D. Power and delay efficient multilevel offloading strategies for mobile cloud computing. Wireless Personal Communications. 2020. vol. 112. pp. 2159–2186.

16. Sun M., Xu X., Tao X., Zhang P. Large-scale user-assisted multi-task online offloading for latency reduction in D2D-enabled heterogeneous networks. IEEE Transactions on Network Science and Engineering. 2020. vol. 7. no. 4. pp. 2456–2467.

17. Niu H., Wang L., Du K., Lu Z., Wen X., Liu Y. A pipelining task offloading strategy via delay-aware multi-agent reinforcement learning in Cybertwin-enabled 6G network. Digital Communications and Networks. 2023. DOI: 10.1016/j.dcan.2023.04.004.

18. Liu H., Niu Z., Du J., Lin X. Genetic algorithm for delay efficient computation offloading in dispersed computing. Ad Hoc Networks. 2023. vol. 142. no. 103109.

19. Mirza M.A., Yu J., Raza S., Krichen M., Ahmed M., Khan W.U., Rabie K., Shongwe T. DRL-assisted delay optimized task offloading in Automotive-Industry 5.0 based VECNs. Journal of King Saud University-Computer and Information Sciences. 2023. vol. 35(6). no. 101512. DOI: 10.1016/j.jksuci.2023.02.013.

20. Li X., Ye B. Latency-Aware Computation Offloading for 5G Networks in Edge Computing. Security and Communication Networks. 2021. vol. 2021. pp. 1–15.

21. Cozzolino V., Tonetto L., Mohan N., Ding A.Y., Ott J. Nimbus: Towards latency-energy efficient task offloading for ar services. IEEE Transactions on Cloud Computing. 2023. vol. 11. no. 2. pp. 1530–1545. DOI: 10.1109/TCC.2022.3146615.

22. Liu C.F., Bennis M., Debbah M., Poor H.V. Dynamic task offloading and resource allocation for ultra-reliable low-latency edge computing. IEEE Transactions on Communications. 2019. vol. 67. no. 6. pp. 4132–4150.

23. Zhang H., Yang Y., Huang X., Fang C., Zhang P. Ultra-low latency multi-task offloading in mobile edge computing. IEEE Access, 2021. vol. 9. pp. 32569–32581.

24. Yang T., Feng H., Gao S., Jiang Z., Qin M., Cheng N., Bai L. Two-stage offloading optimization for energy–latency tradeoff with mobile edge computing in maritime Internet of Things. IEEE Internet of Things Journal. 2019. vol. 7. no. 7. pp. 5954–5963.

25. Shu C., Zhao Z., Han Y., Min G., Duan H. Multi-user offloading for edge computing networks: A dependency-aware and latency-optimal approach. IEEE Internet of Things Journal. 2019. vol. 7. no. 3. pp. 1678–1689.
26. Gu X., Ji C., Zhang G. Energy-optimal latency-constrained application offloading in mobile-edge computing. Sensors. 2020. vol. 20(11). no. 3064.
27. Liu S., Yu Y., Guo L., Yeoh P.L., Vucetic B., Li Y. Adaptive delay-energy balanced partial offloading strategy in Mobile Edge Computing networks. Digital Communications and Networks. 2022. DOI: 10.1016/j.dcan.2022.05.029.
28. Zhang Y., Chen J., Zhou Y., Yang L., He B., Yang Y. Dependent task offloading with energy-latency tradeoff in mobile edge computing. IET Communications. 2022. vol. 16. no. 17. pp. 1993–2001.
29. Li Y., Wang T., Wu Y., Jia W. Optimal dynamic spectrum allocation-assisted latency minimization for multiuser mobile edge computing. Digital Communications and Networks. 2022. vol. 8. no. 3. pp. 247–256.
30. Wang M., Wu T., Ma T., Fan X., Ke M. Users' experience matter: Delay sensitivity-aware computation offloading in mobile edge computing. Digital Communications and Networks. 2022. vol. 8. no. 6. pp. 955–963.
31. Elgendy I.A., Zhang W.Z., Liu C.Y., Hsu C.H. An efficient and secured framework for mobile cloud computing. IEEE Transactions on Cloud Computing. 2018. vol. 9. no. 1. pp. 79–87.
32. Tyagi H., Kumar R. Cloud computing for IoT. Internet of Things (IoT) Concepts and Applications. 2020. pp. 25–41.
33. Cong P., Zhou J., Li L., Cao K., Wei T., Li K. A survey of hierarchical energy optimization for mobile edge computing: A perspective from end devices to the cloud. ACM Computing Surveys (CSUR). 2020. vol. 53. no. 2. pp. 1–44.
34. Elgendy I.A., Zhang W., Tian Y.C., Li K. Resource allocation and computation offloading with data security for mobile edge computing. Future Generation Computer Systems. 2019. vol. 100. pp. 531–541.
35. Zhang W.Z., Elgendy I.A., Hammad M., Iliyasu A.M., Du X., Guizani M., Abd el-Latif A.A. Secure and optimized load balancing for multitier IoT and edge-cloud computing systems. IEEE Internet of Things Journal. 2021. vol. 8. no. 10. pp. 8119–8132.
36. Elgendy I.A., Zhang W.Z., Zeng Y., He H., Tian Y.C., Yang Y. Efficient and secure multi-user multi-task computation offloading for mobile-edge computing in mobile IoT networks. IEEE Transactions on Network and Service Management. 2020. vol. 17. no. 4. pp. 2410–2422.
37. Mahmud R., Ramamohanarao K., Buyya R. Application management in fog computing environments: A taxonomy, review and future directions. ACM Computing Surveys (CSUR). 2020. vol. 53. no. 4. pp. 1–43.
38. Helbig M., Deb K., Engelbrecht A. Key challenges and future directions of dynamic multi-objective optimisation. IEEE Congress on Evolutionary Computation (CEC). IEEE, 2016. pp. 1256–1261.
39. Almutairi J., Aldossary M. A novel approach for IoT tasks offloading in edge-cloud environments. Journal of Cloud Computing. 2021. vol. 10(1). pp. 1–19.
40. Almutairi J., Aldossary M., Alharbi H.A., Yosuf B.A., Elmirghani J.M. Delay-optimal task offloading for UAV-enabled edge-cloud computing systems. IEEE Access. 2022. vol. 10. pp. 51575–51586.
41. Wang Y., Wang L., Zheng R., Zhao X., Liu M. Latency-optimal computational offloading strategy for sensitive tasks in smart homes. Sensors. 2021. vol. 21(7). no. 2347.

42. Ren J., Yu G., He Y., Li G.Y. Collaborative cloud and edge computing for latency minimization. IEEE Transactions on Vehicular Technology. 2019. vol. 68. no. 5. pp. 5031–5044.
43. Lakhan A., Mohammed M.A., Abdulkareem K.H., Jaber M.M., Nedoma J., Martinek R., Zmij P. Delay optimal schemes for Internet of Things applications in heterogeneous edge cloud computing networks. Sensors. 2022. vol. 22(16). no. 5937.
44. AlShathri S.I., Hassan D.S., Chelloug S.A. Latency-Aware Dynamic Second Offloading Service in SDN-Based Fog Architecture. CMC-Computers Materials and Continua. 2023. vol. 75. no. 1. pp. 1501–1526.
45. Kaur P., Mehta S. Improvement of Task Offloading for Latency Sensitive Tasks in Fog Environment. Energy Conservation Solutions for Fog-Edge Computing Paradigms. 2022. pp. 49–63.
46. Mukherjee M., Kumar V., Kumar S., Matamy R., Mavromoustakis C.X., Zhang Q., Shojafar M., Mastorakis G. Computation offloading strategy in heterogeneous fog computing with energy and delay constraints. IEEE International Conference on Communications (ICC). IEEE. 2020. pp. 1–5. DOI: 10.1109/ICC40277.2020.9148852.
47. Tran-Dang H., Kim D.S. Dynamic collaborative task offloading for delay minimization in the heterogeneous fog computing systems. Journal of Communications and Networks. 2023. vol. 25. no. 2. pp. 244–252. DOI: 10.23919/JCN.2023.000008.
48. Tran-Dang H., Kim D.S. FRATO: Fog resource based adaptive task offloading for delay-minimizing IoT service provisioning. IEEE Transactions on Parallel and Distributed Systems. 2021. vol. 32. no. 10. pp. 2491–2508.
49. Kishor A., Chakarbarty C. Task offloading in fog computing for using smart ant colony optimization. Wireless personal communications. 2021. pp. 1–22.
50. Ren Q., Liu K., Zhang L. Multi-objective optimization for task offloading based on network calculus in fog environments. Digital Communications and Networks. 2022. vol. 8(5). pp. 825–833.
51. Tran-Dang H., Kim D.S. Distributed Computation Offloading Framework for Fog Computing Networks. Cooperative and Distributed Intelligent Computation in Fog Computing: Concepts, Architectures, and Frameworks. 2023. pp. 133–155.
52. Chakraborty C., Mishra K., Majhi S.K., Bhuyan H.K. Intelligent Latency-aware tasks prioritization and offloading strategy in Distributed Fog-Cloud of Things. IEEE Transactions on Industrial Informatics. 2022. vol. 19(2). pp. 2099–2106.
53. Cui K., Lin B., Sun W., Sun W. Learning-based task offloading for marine fog-cloud computing networks of USV cluster. Electronics. 2019. vol. 8(11). no. 1287.
54. Bukhari M.M., Ghazal T.M., Abbas S., Khan M.A., Farooq U., Wahbah H., Ahmad M., Adnan, K M. An intelligent proposed model for task offloading in fog-cloud collaboration using logistics regression. Computational Intelligence and Neuroscience. 2022. vol. 2022. DOI: 10.1155/2022/3606068.
55. Pan Y., Jiang H., Zhu H., Wang J. Latency minimization for task offloading in hierarchical fog-computing C-RAN networks. IEEE International Conference on Communications (ICC). IEEE, 2020. pp. 1–6.
56. Mahini H., Rahmani A.M., Mousavirad S.M. An evolutionary game approach to IoT task offloading in fog-cloud computing. The Journal of Supercomputing. 2021. vol. 77. pp. 5398–5425.
57. Jindal R., Kumar N., Nirwan H. MTFCT: A task offloading approach for fog computing and cloud computing. 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence). IEEE, 2020. pp. 145–149.

58. Jain V., Kumar B. Optimal task offloading and resource allotment towards fog-cloud architecture. 11th International Conference on Cloud Computing, Data Science and Engineering (Confluence). IEEE. 2021. pp. 233–238.
59. Guo M., Li L., Guan Q. Energy-efficient and delay-guaranteed workload allocation in IoT-edge-cloud computing systems. IEEE Access. 2019. vol. 7. pp. 78685–78697.
60. Wu H., Wolter K., Jiao P., Deng Y., Zhao Y., Xu M. EEDTO: An energy-efficient dynamic task offloading algorithm for blockchain-enabled IoT-edge-cloud orchestrated computing. IEEE Internet of Things Journal. 2020. vol. 8. no. 4. pp. 2163–2176.
61. Hong Z., Chen W., Huang H., Guo S., Zheng Z. Multi-hop cooperative computation offloading for industrial IoT–edge–cloud computing environments. IEEE Transactions on Parallel and Distributed Systems. 2019. vol. 30. no. 12. pp. 2759–2774.
62. Qu G., Wu H., Li R., Jiao P. DMRO: A deep meta reinforcement learning-based task offloading framework for edge-cloud computing. IEEE Transactions on Network and Service Management. 2021. vol. 18. no. 3. pp. 3448–3459.
63. Gali M., Mahamkali A. A Distributed Deep Meta Learning based Task Offloading Framework for Smart City Internet of Things with Edge-Cloud Computing. Journal of Internet Services and Information Security. 2022. vol. 12. no. 4. pp. 224–237.

**Swapna B** — Ph.D., Employee, Koneru Lakshmaiah Education Foundation (Deemed to be University), Vaddeshwaram. Research interests: computer science and engineering. The number of publications — 2. swapnaswapb@gmail.com; Green Fields, Vaddeswaram, 522302, Guntur, India; office phone: +91(8645)350-0200.

**Divya V** — Ph.D., Dr.Sci., Associate professor, Koneru Lakshmaiah Education Foundation (Deemed to be University), Vaddeshwaram. Research interests: network security, cloud computing. The number of publications — 6. divya.movva@kluniversity.in; Green Fields, Vaddeswaram, 522302, Guntur, Russia; office phone: +91(8645)350-0200.

**УДК** 004.7 **DOI** 10.15622/ia.23.1.10

Б. Свапна, В. Дивья

# ИНТЕЛЛЕКТУАЛЬНАЯ СХЕМА РАСПРЕДЕЛЕНИЕ ЗАДАЧ С УЧЕТОМ ЗАДЕРЖЕК ВЫЧИСЛЕНИЙ В EDGE-FOG-CLOUD – ОБЗОР

*Свапна Б., Дивья В.* **Интеллектуальная схема распределения задач с учетом задержек вычислений в Edge-Fog-Cloud – обзор.**

**Аннотация.** Огромный объем данных, создаваемых процедурами Интернета вещей, требует вычислительной мощности и места для хранения, предоставляемого облачными, периферийными и туманными вычислительными системами. Каждый из этих способов вычислений имеет как преимущества, так и недостатки. Облачные вычисления улучшают хранение информации и вычислительные возможности, одновременно увеличивая задержку соединения. Периферийные и туманные вычисления предлагают аналогичные преимущества с уменьшенной задержкой, но имеют ограниченное хранилище, емкость и покрытие. Первоначально оптимизация применялась для решения проблемы сброса трафика. И наоборот, традиционная оптимизация не может удовлетворить жесткие требования к задержке принятия решений в сложных системах, варьирующейся от миллисекунд до долей секунды. В результате алгоритмы машинного обучения, особенно обучение с подкреплением, набирают популярность, поскольку они могут быстро решать проблемы разгрузки в динамических ситуациях, включающих определенные неопознанные данные. Мы проводим анализ литературы, чтобы изучить различные методы, используемые для решения этой интеллектуальной задачи по разгрузке задач с учетом задержек для облачных, периферийных и туманных вычислений. Уроки, полученные в результате этих исследований, затем представлены в настоящем отчете. Наконец, мы определяем некоторые дополнительные возможности для изучения и проблемы, которые необходимо преодолеть, чтобы достичь минимальной задержки в системе разгрузки задач.

**Ключевые слова:** разгрузка задач, облачные вычисления, периферийные вычисления, туманные вычисления, Интернет вещей, задержка.

## Литература

1. Wang F., Zhu M., Wang M., Khosravi M.R., Ni Q., Yu S., Qi L. 6G-enabled short-term forecasting for large-scale traffic flow in massive IoT based on time-aware locality-sensitive hashing. IEEE Internet of Things Journal. 2020. vol. 8. no. 7. pp. 5321–5331.
2. Wei D., Ning H., Shi F., Wan Y., Xu J., Yang S., Zhu L. Dataflow management in the internet of things: Sensing, control, and security. Tsinghua Science and Technology. 2021. vol. 26. no. 6. pp. 918–930.
3. Zheng T., Wan J., Zhang J., Jiang C., Jia G. A survey of computation offloading in edge computing. In 2020 International Conference on Computer, Information and Telecommunication Systems (CITS). IEEE, 2020. pp. 1–6.
4. Saeik F., Avgeris M., Spatharakis D., Santi N., Dechouniotis D., Violos J., Papavassiliou S. Task offloading in Edge and Cloud Computing: A survey on mathematical, artificial intelligence and control theory solutions. Computer Networks. 2021. vol. 195. no.108177.
5. Zhao T., Zhou S., Guo X., Zhao Y., Niu Z. A cooperative scheduling scheme of local cloud and internet cloud for delay-aware mobile cloud computing. IEEE globecom workshops (GC Wkshps). IEEE, 2015. pp. 1–6.

6. Xu F., Yang W., Li H. Computation offloading algorithm for cloud robot based on improved game theory. Computers & Electrical Engineering. 2020. vol. 87. no. 106764.
7. Shakarami A., Ghobaei-Arani M., Masdari M., Hosseinzadeh M. A survey on the computation offloading approaches in mobile edge/cloud computing environment: a stochastic-based perspective. Journal of Grid Computing. 2020. vol. 18. pp. 639–671.
8. Guo S., Zeng D., Gu L., Luo J. When green energy meets cloud radio access network: Joint optimization towards brown energy minimization. Mobile Networks and Applications. 2019. vol. 24. pp. 962–970.
9. Dai H.N., Wong R.C.W., Wang H., Zheng Z., Vasilakos A.V. Big data analytics for large-scale wireless networks: Challenges and opportunities. ACM Computing Surveys (CSUR). 2019. vol. 52. no. 5. pp. 1–36.
10. Hong C.H., Varghese, B. Resource management in fog/edge computing: a survey on architectures, infrastructure, and algorithms. ACM Computing Surveys (CSUR). 2019. vol. 52. no. 5. pp. 1–36.
11. Xu Z., Liang W., Jia M., Huang M., Mao G. Task offloading with network function requirements in a mobile edge-cloud network. IEEE Transactions on Mobile Computing. 2018. vol. 18. no. 11. pp. 2672–2685.
12. Ren J., Zhang D., He S., Zhang Y., Li T. A survey on end-edge-cloud orchestrated network computing paradigms: Transparent computing, mobile edge computing, fog computing, and cloudlet. ACM Computing Surveys (CSUR). 2019. vol. 52. no. 6. pp. 1–36.
13. Zhang Z., Li C., Peng S., Pei X. A new task offloading algorithm in edge computing. EURASIP Journal on Wireless Communications and Networking. 2021. vol. 2021. pp. 1–21.
14. You C., Huang K., Chae H., Kim B.H. Energy-efficient resource allocation for mobile-edge computation offloading. IEEE Transactions on Wireless Communications. 2016. vol. 16. no. 3. pp. 1397–1411.
15. De D., Mukherjee A., Guha Roy D. Power and delay efficient multilevel offloading strategies for mobile cloud computing. Wireless Personal Communications. 2020. vol. 112. pp. 2159–2186.
16. Sun M., Xu X., Tao X., Zhang P. Large-scale user-assisted multi-task online offloading for latency reduction in D2D-enabled heterogeneous networks. IEEE Transactions on Network Science and Engineering. 2020. vol. 7. no. 4. pp. 2456–2467.
17. Niu H., Wang L., Du K., Lu Z., Wen X., Liu Y. A pipelining task offloading strategy via delay-aware multi-agent reinforcement learning in Cybertwin-enabled 6G network. Digital Communications and Networks. 2023. DOI: 10.1016/j.dcan.2023.04.004.
18. Liu H., Niu Z., Du J., Lin X. Genetic algorithm for delay efficient computation offloading in dispersed computing. Ad Hoc Networks. 2023. vol. 142. no. 103109.
19. Mirza M.A., Yu J., Raza S., Krichen M., Ahmed M., Khan W.U., Rabie K., Shongwe T. DRL-assisted delay optimized task offloading in Automotive-Industry 5.0 based VECNs. Journal of King Saud University-Computer and Information Sciences. 2023. vol. 35(6). no. 101512. DOI: 10.1016/j.jksuci.2023.02.013.
20. Li X., Ye B. Latency-Aware Computation Offloading for 5G Networks in Edge Computing. Security and Communication Networks. 2021. vol. 2021. pp. 1–15.
21. Cozzolino V., Tonetto L., Mohan N., Ding A.Y., Ott J. Nimbus: Towards latency-energy efficient task offloading for ar services. IEEE Transactions on Cloud Computing. 2023. vol. 11. no. 2. pp. 1530–1545. DOI: 10.1109/TCC.2022.3146615.

22. Liu C.F., Bennis M., Debbah M., Poor H.V. Dynamic task offloading and resource allocation for ultra-reliable low-latency edge computing. IEEE Transactions on Communications. 2019. vol. 67. no. 6. pp. 4132–4150.
23. Zhang H., Yang Y., Huang X., Fang C., Zhang P. Ultra-low latency multi-task offloading in mobile edge computing. IEEE Access, 2021. vol. 9. pp. 32569–32581.
24. Yang T., Feng H., Gao S., Jiang Z., Qin M., Cheng N., Bai L. Two-stage offloading optimization for energy–latency tradeoff with mobile edge computing in maritime Internet of Things. IEEE Internet of Things Journal. 2019. vol. 7. no. 7. pp. 5954–5963.
25. Shu C., Zhao Z., Han Y., Min G., Duan H. Multi-user offloading for edge computing networks: A dependency-aware and latency-optimal approach. IEEE Internet of Things Journal. 2019. vol. 7. no. 3. pp. 1678–1689.
26. Gu X., Ji C., Zhang G. Energy-optimal latency-constrained application offloading in mobile-edge computing. Sensors. 2020. vol. 20(11). no. 3064.
27. Liu S., Yu Y., Guo L., Yeoh P.L., Vucetic B., Li Y. Adaptive delay-energy balanced partial offloading strategy in Mobile Edge Computing networks. Digital Communications and Networks. 2022. DOI: 10.1016/j.dcan.2022.05.029.
28. Zhang Y., Chen J., Zhou Y., Yang L., He B., Yang Y. Dependent task offloading with energy-latency tradeoff in mobile edge computing. IET Communications. 2022. vol. 16. no. 17. pp. 1993–2001.
29. Li Y., Wang T., Wu Y., Jia W. Optimal dynamic spectrum allocation-assisted latency minimization for multiuser mobile edge computing. Digital Communications and Networks. 2022. vol. 8. no. 3. pp. 247–256.
30. Wang M., Wu T., Ma T., Fan X., Ke M. Users' experience matter: Delay sensitivity-aware computation offloading in mobile edge computing. Digital Communications and Networks. 2022. vol. 8. no. 6. pp. 955–963.
31. Elgendy I.A., Zhang W.Z., Liu C.Y., Hsu C.H. An efficient and secured framework for mobile cloud computing. IEEE Transactions on Cloud Computing. 2018. vol. 9. no. 1. pp. 79–87.
32. Tyagi H., Kumar R. Cloud computing for IoT. Internet of Things (IoT) Concepts and Applications. 2020. pp. 25–41.
33. Cong P., Zhou J., Li L., Cao K., Wei T., Li K. A survey of hierarchical energy optimization for mobile edge computing: A perspective from end devices to the cloud. ACM Computing Surveys (CSUR). 2020. vol. 53. no. 2. pp. 1–44.
34. Elgendy I.A., Zhang W., Tian Y.C., Li K. Resource allocation and computation offloading with data security for mobile edge computing. Future Generation Computer Systems. 2019. vol. 100. pp. 531–541.
35. Zhang W.Z., Elgendy I.A., Hammad M., Iliyasu A.M., Du X., Guizani M., Abd el-Latif A.A. Secure and optimized load balancing for multitier IoT and edge-cloud computing systems. IEEE Internet of Things Journal. 2021. vol. 8. no. 10. pp. 8119–8132.
36. Elgendy I.A., Zhang W.Z., Zeng Y., He H., Tian Y.C., Yang Y. Efficient and secure multi-user multi-task computation offloading for mobile-edge computing in mobile IoT networks. IEEE Transactions on Network and Service Management. 2020. vol. 17. no. 4. pp. 2410–2422.
37. Mahmud R., Ramamohanarao K., Buyya R. Application management in fog computing environments: A taxonomy, review and future directions. ACM Computing Surveys (CSUR). 2020. vol. 53. no. 4. pp. 1–43.
38. Helbig M., Deb K., Engelbrecht A. Key challenges and future directions of dynamic multi-objective optimisation. IEEE Congress on Evolutionary Computation (CEC). IEEE, 2016. pp. 1256–1261.

39. Almutairi J., Aldossary M. A novel approach for IoT tasks offloading in edge-cloud environments. Journal of Cloud Computing. 2021. vol. 10(1). pp. 1–19.
40. Almutairi J., Aldossary M., Alharbi H.A., Yosuf B.A., Elmirghani J.M. Delay-optimal task offloading for UAV-enabled edge-cloud computing systems. IEEE Access. 2022. vol. 10. pp. 51575–51586.
41. Wang Y., Wang L., Zheng R., Zhao X., Liu M. Latency-optimal computational offloading strategy for sensitive tasks in smart homes. Sensors. 2021. vol. 21(7). no. 2347.
42. Ren J., Yu G., He Y., Li G.Y. Collaborative cloud and edge computing for latency minimization. IEEE Transactions on Vehicular Technology. 2019. vol. 68. no. 5. pp. 5031–5044.
43. Lakhan A., Mohammed M.A., Abdulkareem K.H., Jaber M.M., Nedoma J., Martinek R., Zmij P. Delay optimal schemes for Internet of Things applications in heterogeneous edge cloud computing networks. Sensors. 2022. vol. 22(16). no. 5937.
44. AlShathri S.I., Hassan D.S., Chelloug S.A. Latency-Aware Dynamic Second Offloading Service in SDN-Based Fog Architecture. CMC-Computers Materials and Continua. 2023. vol. 75. no. 1. pp. 1501–1526.
45. Kaur P., Mehta S. Improvement of Task Offloading for Latency Sensitive Tasks in Fog Environment. Energy Conservation Solutions for Fog-Edge Computing Paradigms. 2022. pp. 49–63.
46. Mukherjee M., Kumar V., Kumar S., Matamy R., Mavromoustakis C.X., Zhang Q., Shojafar M., Mastorakis G. Computation offloading strategy in heterogeneous fog computing with energy and delay constraints. IEEE International Conference on Communications (ICC). IEEE. 2020. pp. 1–5. DOI: 10.1109/ICC40277.2020.9148852.
47. Tran-Dang H., Kim D.S. Dynamic collaborative task offloading for delay minimization in the heterogeneous fog computing systems. Journal of Communications and Networks. 2023. vol. 25. no. 2. pp. 244–252. DOI: 10.23919/JCN.2023.000008.
48. Tran-Dang H., Kim D.S. FRATO: Fog resource based adaptive task offloading for delay-minimizing IoT service provisioning. IEEE Transactions on Parallel and Distributed Systems. 2021. vol. 32. no. 10. pp. 2491–2508.
49. Kishor A., Chakarbarty C. Task offloading in fog computing for using smart ant colony optimization. Wireless personal communications. 2021. pp. 1–22.
50. Ren Q., Liu K., Zhang L. Multi-objective optimization for task offloading based on network calculus in fog environments. Digital Communications and Networks. 2022. vol. 8(5). pp. 825–833.
51. Tran-Dang H., Kim D.S. Distributed Computation Offloading Framework for Fog Computing Networks. Cooperative and Distributed Intelligent Computation in Fog Computing: Concepts, Architectures, and Frameworks. 2023. pp. 133–155.
52. Chakraborty C., Mishra K., Majhi S.K., Bhuyan H.K. Intelligent Latency-aware tasks prioritization and offloading strategy in Distributed Fog-Cloud of Things. IEEE Transactions on Industrial Informatics. 2022. vol. 19(2). pp. 2099–2106.
53. Cui K., Lin B., Sun W., Sun W. Learning-based task offloading for marine fog-cloud computing networks of USV cluster. Electronics. 2019. vol. 8(11). no. 1287.
54. Bukhari M.M., Ghazal T.M., Abbas S., Khan M.A., Farooq U., Wahbah H., Ahmad M., Adnan, K M. An intelligent proposed model for task offloading in fog-cloud collaboration using logistics regression. Computational Intelligence and Neuroscience. 2022. vol. 2022. DOI: 10.1155/2022/3606068.
55. Pan Y., Jiang H., Zhu H., Wang J. Latency minimization for task offloading in hierarchical fog-computing C-RAN networks. IEEE International Conference on Communications (ICC). IEEE, 2020. pp. 1–6.

56. Mahini H., Rahmani A.M., Mousavirad S.M. An evolutionary game approach to IoT task offloading in fog-cloud computing. The Journal of Supercomputing. 2021. vol. 77. pp. 5398–5425.
57. Jindal R., Kumar N., Nirwan H. MTFCT: A task offloading approach for fog computing and cloud computing. 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence). IEEE, 2020. pp. 145–149.
58. Jain V., Kumar B. Optimal task offloading and resource allotment towards fog-cloud architecture. 11th International Conference on Cloud Computing, Data Science and Engineering (Confluence). IEEE. 2021. pp. 233–238.
59. Guo M., Li L., Guan Q. Energy-efficient and delay-guaranteed workload allocation in IoT-edge-cloud computing systems. IEEE Access. 2019. vol. 7. pp. 78685–78697.
60. Wu H., Wolter K., Jiao P., Deng Y., Zhao Y., Xu M. EEDTO: An energy-efficient dynamic task offloading algorithm for blockchain-enabled IoT-edge-cloud orchestrated computing. IEEE Internet of Things Journal. 2020. vol. 8. no. 4. pp. 2163–2176.
61. Hong Z., Chen W., Huang H., Guo S., Zheng Z. Multi-hop cooperative computation offloading for industrial IoT–edge–cloud computing environments. IEEE Transactions on Parallel and Distributed Systems. 2019. vol. 30. no. 12. pp. 2759–2774.
62. Qu G., Wu H., Li R., Jiao P. DMRO: A deep meta reinforcement learning-based task offloading framework for edge-cloud computing. IEEE Transactions on Network and Service Management. 2021. vol. 18. no. 3. pp. 3448–3459.
63. Gali M., Mahamkali A. A Distributed Deep Meta Learning based Task Offloading Framework for Smart City Internet of Things with Edge-Cloud Computing. Journal of Internet Services and Information Security. 2022. vol. 12. no. 4. pp. 224–237.

**Свапна Б** — Ph.D., сотрудник, Образовательный фонд Конеру Лакшмайи (Считается университетом). Область научных интересов: информатика и инженерия. Число научных публикаций — 2. swapnaswapb@gmail.com; Зеленые поля, Ваддесварам, 522302, Гунтур, Индия; р.т.: +91(8645)350-0200.

**Дивья В** — Ph.D., Dr.Sci., доцент, Образовательный фонд Конеру Лакшмайи (Считается университетом). Область научных интересов: сетевая безопасность, облачные вычисления. Число научных публикаций — 6. divya.movva@kluniversity.in; Зеленые поля, Ваддесварам, 522302, Гунтур, Россия; р.т.: +91(8645)350-0200.