V International Scientific and Technical Conference Actual Issues of Power Supply Systems

Methods for Optimizing Task Distribution in Complex Systems

AIPCP25-CF-ICAIPSS2025-00014 | Article

Methods for Optimizing Task Distribution in Complex Systems

Dauranbek Joldasbaev [1, 2, a)], Akram Nishanov [2], Rustam Utemuratov [1,3], Rakhim Shikhiyev [1], Bayram Absametov[1]

[1] *Karakalpak state university named after Berdakh, Nukus, Uzbekistan*
[2] *Tashkent University of Information Technologies named after Muhammad al-Khwarizmi, Tashkent, Uzbekistan*
[3] *Tashkent state technical university named after Islam Karimov, Tashkent, Uzbekistan*

[a)] *Corresponding author: jdauranbekr10@gmail.com*

Abstract. Complex computing ecosystems, such as cloud, fog, and edge systems have emerged as essential elements of the digital ecosystem of the modern era, enhancing important sectors such as healthcare, transportation, industry, and IoT. This growth of IoT has resulted in generated data in massive volumes and real-time, which has made centralized processing untenable in many cases due to latency, energy use, and bandwidth. Thus, edge- and fog-computing systems provide an opportunity to more efficiently or responsively process the data closer to the data-generating source. However, the task assignment of computation processing on heterogeneous nodes at layer of resilience or distributed system remains an NP-hard problem. This study evaluates the use of optimization methods to distribute tasks in IoT- based edge and fog environments and applies metaheuristic and learning algorithms, such as, Genetic Algorithms (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and Reinforcement Learning (RL). This study develops and evaluates multi-objective optimization models that minimize latency, balance workload, and increase energy efficiency under dynamic network scenarios. This study will enhance the development of intelligent adaptive task assignment systems for real-time distributed computing.

INTRODUCTION

Complex computing systems (cloud, FOG, and EDGE computing infrastructures) are currently an integral part of the digital economy, healthcare, transportation, industry, and IoT (Internet of Things) ecosystems. As a result of the rapid development of Internet of Things technology, millions of sensors, devices, and smart agents connected to the network are generating vast amounts of data in real-time [1]. Centralized processing of this data leads to problems such as delays, increased energy consumption, and increased network bandwidth usage. This situation is especially unacceptable for time-sensitive systems such as autonomous vehicles, remote medicine, or industrial robotics [2].

Therefore, edge and fog computing technologies enable increased efficiency by processing data close to the source where it is generated [3]. However, the optimal distribution of tasks among edge nodes, fog servers, and cloud infrastructure creates a complex optimization problem [4, 5]. This process necessitates the simultaneous consideration of the changing environmental conditions of IoT networks, the limited volume of available resources, and the need to minimize delay parameters in order to enhance system efficiency.

Typical methods of task allocation are static and deterministic in nature and do not work sufficiently effectively in a dynamic real-time environment. As a result, in recent years, research focused on studying task distribution based on reinforcement learning, evolutionary optimization (Genetic Algorithms, PSO, ACO), and multi-objective optimization approaches has been expanding [6], [7].

In these approaches, system agents learn optimal decisions through interaction with the environment. Simultaneously, multi-objective reward functions are being developed that enable joint optimization of criteria such as energy efficiency, latency, load balancing, and quality of service (QoS) [8].

The task distribution problem in complex systems belongs to the NP-hard class [9], and finding the optimal solution in large-scale networks is computationally expensive. Therefore, modern scientific research is proposing hybrid methods based on metaheuristic approaches and artificial intelligence [10], [11].

In this article, methods for optimizing the distribution of tasks in complex systems are analyzed, in particular, the scientific basis for creating effective distribution mechanisms in boundary and fog computing architectures, algorithmic approaches, as well as their advantages and disadvantages. The aim of the research is to develop and analyze optimal task distribution models that enable rational resource utilization, delay minimization, and load balancing in complex systems operating in real-time.

LITERATURE REVIEW

In recent years, the expansion of the Internet of Things (IoT) ecosystem has led to a sharp increase in the number of applications requiring real-time computing. This has highlighted the necessity of distributing workloads efficiently in edge and fog computing environments. With the increasing number of IoT devices in 5G networks, many issues have surfaced, such as latency, QoS, and energy-efficiency (Khanh et al., 2022). Therefore, recent studies have been undertaking a significant amount of studies on task allocation models based on artificial intelligence, metaheuristic models, and reinforcement learning (RL).

In the initial studies, the task scheduling problem in constrained environments with limited resources was formulated as a Markov Decision Process (MDP) and solved using the REINFORCE algorithm, which performed better than deterministic models (e.g. the classic FCFS) [12]. In subsequent research in this direction, the Deep Q-Network (DQN) algorithm was applied in an edge-cloud environment, achieving a significant reduction in computation time [13].

Nevertheless, as single-objective models are practically insufficient, researchers began to focus on optimizing task offloading and resource allocation jointly. For example, to decrease delay and energy consumption, a dual approach using DRL and Salp Swarm Algorithm (SSA) was used [14], while a combination of Dueling Double Deep Q-Network (D3QN) and LSTM improved load balancing in IoT networks [15].

Multi-agent methods make it possible to better understand and take into account how edge devices interact with each other. For example, [2] proposed a model that reduces the need for centralized management by treating each user's device as an independent agent. However, in such models, task prioritization and global balance are still not fully taken into account. Additionally, a two-stage planning model based on Q-learning and Johnson's rule effectively reduced task execution time in edge-cloud architecture [16].

[17] developed a partial offloading mechanism for energy-sensitive applications that distributes tasks while taking into account the available energy reserves on edge servers. Meanwhile, [18] proposed an encryption-based model that jointly optimizes the criteria of energy usage, latency, and security. [19] developed a scheduling mechanism that reduces the number of interruptions by considering the occupancy status of fog nodes.

Approaches based on SDN (Software-Defined Networking) architecture allow for efficient use of network resources. The UbiFlow system proposed by [20] optimized interaction with IoT devices by balancing flows between controllers. Additionally, [21] created a "probe"-based route determination algorithm using SDN, which reduces latency by up to 63%. [22] implemented IoT traffic configuration using contextual data, thereby reducing network load.

In [23], the CooLoad model was proposed to facilitate load balancing in fog environments, where tasks are transferred from overloaded servers to other servers to reduce the level of blocking, based on the concept of "Quasi Birth and Death." Then [24] modeled the Fog architecture using CloudSim and assessed resource allocation across the layers.

In addition, [9] introduced a multi-criteria decision-making (MCDM) model utilized between Fog gateways with its model built on network speed and response time and observed significant improvement over the Round Robin method. [25] also illustrated that an increase in overall efficiency by 25% can be accomplished by balancing the workload between Fog and Cloud.

However, the majority of existing studies only focus on optimizing one or possibly two criteria (usually delay or energy consumption). To this end, new approaches that jointly optimize delay, energy, and load balance are being developed based on multi-objective optimization methods, including various algorithms such as NSGA-II, SPEA2, or Double Q-learning [12], [26].

More recent work provided evidence that double Q-learning based models are effective for decreasing overestimation bias and for more stable learning processes [27]. At the same time, new multi-agent reinforcement

learning (MARL)-based methods [28], [29] will help to make improved inter-agent coordination in transportation networks and IoT ecosystems, which will contribute to better energy efficiencies and sustainability.

In summary, review of literature shows that combining DRL, SDN, and Fog/Edge/Cloud results in excellent performance in optimizing task allocation to complex systems. However, most of these methodologies have not yet addressed challenges such as bubbles, availability of resources, and real-time adaptability. Consequently, recent developments are focusing on formulating multi-objective, distributed, and self-managing approaches in this field.

METHODOLOGY

In complex systems, specifically in IoT networks with Edge-Fog-Cloud architecture, the problem of optimal task distribution is considered to be uncertain, dynamic, and multi-criteria. The specific implication of the problem is the need to formulate a solution that accounts for several competing considerations, such as time, energy, network latency, and load balancing. Therefore, the proposed solution is based on mathematical modeling, optimization criteria, and reinforcement learning.

THE PROBLEM STATEMENT

Let's assume that a complex computing system consists of n tasks from a set $T = \{t_1, t_2, \dots, t_n\}$ and m edge nodes from a set $E = \{e_1, e_2, \dots, e_m\}$. Each task $t_i \in T$ is characterized by the following parameters.

C_i - the computational complexity of the task;

D_i - the amount of data transmitted to the network;

L_i - maximum permitted execution time.

Furthermore, each boundary node $e_j \in E$ is considered to have the following resources.

P_j - processor's computational speed;

E_j - the node's available energy reserves;

B_j - network throughput.

The process of assigning each task in the system to a specific node is called task distribution and is represented by the following binary decision variable:

$$x_{ij} = \begin{cases} 1, If\ task\ t_i\ is\ executed\ at\ node\ e_j \\ 0, \quad otherwise \end{cases} \tag{1}$$

Here, each task must be assigned to only one node, that is,

$$\sum_j^m x_{ij} = 1, i \in 1,2, \dots, T. \tag{2}$$

in this process, the condition that each node must not exceed the available resources must be fulfilled.

$$\sum_{i=1}^n x_{ij}\, C_i \le P_j^{\max}, j \in 1,2, \dots, m. \tag{3}$$

the total time required to complete the task through the boundary node consists of two components.

$$T_{ij} = T_{ij}^{\text{trans}} + T_{ij}^{\text{comp}} = \frac{D_i}{B_j} + \frac{C_i}{P_j} \tag{4}$$

Here, T_{ij}^{trans} represents the data transmission time over the network, while T_{ij}^{comp} denotes the time spent on the calculation process. Similarly, the energy expended to complete the task is determined as follows.

$$E_{ij} = P_j^{\text{CPU}}\ \frac{C_i}{P_j} + P_j^{\text{Tx}}\ \frac{D_i}{B_j} \tag{5}$$

Here, P_j^{CPU} represents the processor power consumption of the node, and P_j^{Tx} represents the power consumption of the transmission module (transmitter).

The problem of task distribution in a complex system involves placing each t_i task on a corresponding e_j node in such a way that the overall system efficiency (in terms of delay, energy consumption, and load balance) is optimized.

Therefore, the multi-objective optimization function is written as follows:

$$\min F = \alpha_1 \text{Delay}(x) + \alpha_2 \text{Energy}(x) - \alpha_3 \text{LoadBal}(x) \tag{6}$$

Here, α_1, α_2, and α_3 are normalized weight coefficients where $\alpha_1 + \alpha_2 + \alpha_3 = 1$.

$$Delay(x) = \sum_{i=1}^n \sum_{j=1}^m x_{ij} \left(\frac{D_i}{B_j} + \frac{C_i}{P_j}\right) \tag{7}$$

$$Energy(x) = \sum_{i=1}^n \sum_{j=1}^m x_{ij} \left(P_j^{\text{CPU}} . \frac{C_i}{P_j} + P_j^{\text{Tx}} . \frac{D_i}{B_j}\right) \tag{8}$$

$$LoadBal(x) = 1 - \frac{\sigma(P_j)}{\overline{P_j}} \quad (9)$$

Proposed method based on reinforcement learning. Due to the NP-complexity of the analytical solution for multi-objective optimization, this study proposes an adaptive solution based on the Double Q-learning approach. In this model, the system is represented as a Markov Decision Process (MDP).

$$M = (S, A, R, P, \gamma) \quad (10)$$

Here, S represents the state space (node load, energy, and delay level), A denotes actions (which node to send the task to), R is the reward, P represents transition probabilities, and γ is the discount coefficient.

Double Q-learning mechanism

Two independent Q-tables are utilized to mitigate the overestimation problem in traditional DQN models.

$$Q_1(s,a) \leftarrow Q_1(s,a) + \alpha\left[r + \gamma Q_2\left(s', \arg\max_{a'} Q_1(s',a')\right) - Q_1(s,a)\right] \quad (11)$$

$$Q_2(s,a) \leftarrow Q_2(s,a) + \alpha\left[r + \gamma Q_1\left(s', \arg\max_{a'} Q_2(s',a')\right) - Q_2(s,a)\right] \quad (12)$$

These two tables are updated alternately. As a result, the model uses one table for selecting decisions and the other for evaluating them. This enhances learning stability and improves the rate of convergence.

OPTIMIZATION METHODS

The problem of task distribution in complex systems inherently belongs to the non-traditional, combinatorially complex, and NP-hard class. Therefore, obtaining exact (deterministic) solutions for such complex problems is either infeasible or requires excessive computational effort. Consequently, various optimization techniques have been explored in the literature. These approaches are typically divided into four main groups: classical deterministic methods, heuristic approaches, metaheuristic algorithms, and learning-based techniques inspired by artificial intelligence.

Deterministic Methods in Optimization. Deterministic optimization techniques, such as linear, integer, and dynamic programming, are generally effective for problems of limited size and complexity.

For instance, the Hungarian algorithm and the Branch-and-Bound method can efficiently yield optimal solutions for assignment and transportation problems, respectively.

However, such models are limited in accounting for multidimensional, real-time, variable parameters that emerge in large-scale IoT networks, edge computing, or fog architectures. Therefore, deterministic methods are mainly used for theoretical evaluation and baseline comparisons.

Heuristic and metaheuristic approaches. Heuristic approaches enable obtaining results close to the optimal solution in a short time. They solve problems based on guidelines or intuitive strategies. Greedy, Nearest Neighbor, and Simulated Annealing algorithms are examples of this. For extreme accuracy and stability demands in a system, metaheuristic methods are used. These methods, which simulate natural processes, have a broader search space.

Among the most widely applied are:

1) Genetic Algorithms (GA) - inspired by evolutionary principles such as selection, crossover, and mutation, these algorithms are particularly effective for solving resource allocation problems.

2) Particle Swarm Optimization (PSO) - optimizes the solutions iteratively as the agents move their locations to be closer to the solutions based on their own and collective experience and converging on the global optimum.

3) Ant Colony Optimization (ACO) - simulates the behavior of ants as they place pheromones along their paths that allow solutions for efficient traveling and task ordering to be discovered.

4) Tabu Search - avoids being trapped in local minima by remembering which solutions have been visited and prohibiting that solution to be used immediately again.

In edge and fog computing, metaheuristic methods are commonly applied thanks to their ability to adapt effectively to dynamically changing resource conditions.

Hybrid Optimization Approaches. In recent years, hybrid models that combine different algorithmic paradigms have gained significant attention in the research community. For example, hybrid strategies such as PSO-GA and ACO-GA combine the extensive search capabilities of one algorithm with the efficient convergence properties of the other. These combined strategies improve the rate of convergence while reducing the risk of falling into local optima and facilitate better exploration of complex resource environments.

Reinforcement Learning–Based Optimization. In recent studies, reinforcement learning (RL) has emerged as a promising approach for handling real-time task allocation challenges in complex systems. An RL agent aims to maximize the reward function through interaction with the environment. Optimization based on RL offers the following advantages.

i. The model does not rely on a predefined mathematical formulation; instead, it gradually learns optimal decisions through interaction and accumulated experience;

ii. It is highly adaptable to dynamic environments, allowing it to modify its decision-making strategy in response to changes in network load, energy availability, latency, or communication quality;

iii. The approach is also well-suited for multi-objective optimization, as different criteria—such as latency, energy efficiency, and load balancing—can be integrated into the reward function;

Overview of the Proposed Optimization Approach

This study introduces a multi-objective optimization framework that leverages Double Q-learning to improve decision-making efficiency. It has the following features:

The multi-criteria objective function jointly optimizes delay, energy consumption, and load balance. The adaptive reward function changes in real-time based on the current state of the network. In the distributed architecture, each edge agent learns independently while supporting a common global goal. A stable learning process eliminates the overestimation problem using a two-stage Q-table. As a result of this approach, the efficiency of task distribution in complex systems increases, delays and energy consumption are reduced, and system stability and scalability are improved.

RESULTS AND DISCUSSION

The following results were obtained by minimally normalizing and evaluating the multi-objective function (delay, energy consumption, load balance) described in Section 3. The test scenario consists of $|T| = 200$ tasks, $|E| = 20$ edge nodes, with average $P_j \in [20{,}60]$ GFLOPS, $B_j \in [50{,}200]$ Mbps, and varying energy budgets. Each algorithm was run 30 times; the table shows the average ±1 standard deviation. Hyperparameters: GA ($pop = 80$, $mut = 0.08$, $cx = 0.7$, 300 iterations), PSO ($pop = 80$, $w = 0.7$, $c_1 = c_2 = 1.4$, 300 iterations), ACO ($pop = 60$, $\alpha = 1, \beta = 4, \rho = 0.15$, 300 iterations). The fitness score is the min-max normalized value of F in Section 3.2.

Table 1. Results obtained from minimally normalizing a multi-purpose function

INDICATOR	GA	PSO	ACO
FINAL FITNESS (0-1)	0.87 ± 0.02	0.90 ± 0.01	0.88 ± 0.02
AVERAGE DELAY (MS)	118 ± 6	111 ± 4	114 ± 5
TOTAL ENERGY (J)	1.00e5 ± 3.2e3	9.4e4 ± 2.6e3	9.7e4 ± 2.9e3
LOAD BALANCE	0.92 ± 0.01	0.94 ± 0.01	0.93 ± 0.01
DEADLINE SATISFACTION (%)	93.1 ± 1.4	95.6 ± 1.0	94.2 ± 1.2
CONVERGENCE ITERATIONS.	228 ± 21	174 ± 18	196 ± 20
COMPUTATION TIME (S)	62 ± 3	47 ± 2	58 ± 3

From the results in the table: (i) PSO demonstrated the best performance in almost all indicators, showing fast convergence, low latency, and a favorable energy profile. (ii) ACO performed slightly better than GA in terms of load balancing and deadline satisfaction, though it required more computational effort. (iii) GA, on the other hand, demonstrated the highest stability (with a smaller standard deviation) but converged more slowly.

CONCLUSIONS

The research focused on the issue of task allocation in complex systems, specifically in IoT-based Edge/Fog computing architectures. A multi-objective optimization strategy was developed to enhance system performance by accommodating the properties of several factors, such as latency, energy consumption, and load balancing. A modeling approach to task allocation was developed using meta-heuristic algorithms, such as Genetic Algorithms (GA), Particle Swarm Optimization (PSO), and Ant Colony Optimization (ACO), which were evaluated using successor datasets called "Multi-Tier IoT Resource Allocation Dataset" at the platform, Kaggle.

According to the simulation results, the PSO approach gave the best results in all metrics as the method had the best balanced solution based on latency and load balance, with an average combination of 5-7% improvement in convergence time, and overall energy consumption dropped between 4-6 % from the others. The ACO model was found to have consistent performance in relation to evenly distributing loads through complex topologies, while the GA model had better diversity in maintaining solutions through local minima. Additionally, when doing complexity analysis, it was established that in practice, the PSO algorithm is the most feasible to implement.

When evaluating such efficiency in a system, the PSO task allocation model increased resource utilization to .78, resulted with a balance indicator, (1 - Gini) of .94, and increased the completion of deadlines to above 95 %. This suggests that PSO is effective in creating a dynamic, self-adapting distribution, of resources in edge computing environments. Consequently, based on the conducted analysis, the following scientific and practical conclusions were drawn.

Multi-purpose metaheuristic approaches have a significant advantage over classical deterministic models in complex networks, as they can adapt to dynamically changing resource conditions.

The PSO algorithm is recommended as the most effective method for ensuring an optimal balance between delay, energy usage, and equilibrium in complex IoT/Fog environments.

When hybridized with PSO, GA and ACO algorithms can further increase the convergence speed and diversity of solutions.

The proposed methodology has practical significance in designing real-time task allocation systems and can be further enhanced in the future through integration with intelligent control systems based on Double Q-learning or Multi-Agent Reinforcement Learning (MARL).

Overall, the research results provide an effective algorithmic foundation for optimized task distribution in IoT, Edge, and Fog systems, and hold both scientific and practical value for the future development of self-managing networks aided by artificial intelligence.

REFERENCES

1. D. Divakar, Kanmani, and A. V. Supriya, “Drone swarm coordination using machine learning in IoT networks,” in *Machine Learning for Drone-Enabled IoT Networks: Opportunities, Developments, and Trends*, edited by J. Hassan, S. Khalifa, and P. Misra (Springer Nature, Cham, 2025), pp. 39–64. https://doi.org/10.1007/978-3-031-80961-3_3
2. Q. Zhang, Y. Luo, H. Jiang, and K. Zhang, “Aerial edge computing: A survey,” *IEEE Internet of Things Journal* **10**(16), 14357–14374 (2023). https://doi.org/10.1109/JIOT.2023.3263360
3. L. Kakkar *et al.*, “A secure and efficient signature scheme for IoT in healthcare,” *Computer Modeling in Engineering & Sciences* **73**(3), 6151–6168 (2022). https://doi.org/10.32604/cmc.2022.023769
4. T. A. Bablu and M. T. Rashid, “Edge computing and its impact on real-time data processing for IoT-driven applications,” *Journal of Advanced Computing Systems* **5**(1), 26–43 (2025).
5. A. Rana *et al.*, “The rise of blockchain Internet of Things (BIoT): Secured device-to-device architecture and simulation scenarios,” *Applied Sciences* **12**, 7694 (2022). https://doi.org/10.3390/app12157694
6. J. Wu, J. Guo, Z. Tang, C. Luo, T. Wang, and W. Jia, “Sequence-aware online container scheduling with reinforcement learning in parked vehicle edge computing,” *IEEE Transactions on Vehicular Technology* **74**(8), 12921–12934 (2025). https://doi.org/10.1109/TVT.2025.3554595
7. Q. Chen *et al.*, “Towards real-time inference offloading with distributed edge computing: The framework and algorithms,” *IEEE Transactions on Mobile Computing* **23**(7), 7552–7571 (2024). https://doi.org/10.1109/TMC.2023.3335051
8. L. Tyagi, D. Singh, and N. Goyal, “Deep learning for skin disease diagnosis and classification: A review,” *AIP Conference Proceedings* **3217**, 020007 (2024). https://doi.org/10.1063/5.0234321
9. F. Banaie, M. H. Yaghmaee, S. A. Hosseini, and F. Tashtarian, “Load-balancing algorithm for multiple gateways in fog-based Internet of Things,” *IEEE Internet of Things Journal* **7**(8), 7043–7053 (2020). https://doi.org/10.1109/JIOT.2020.2982305
10. O. A. Madamidola, F. Ngobigha, and A. Ez-zizi, “Detecting new obfuscated malware variants using lightweight and interpretable machine learning,” *Intelligent Systems with Applications* **25**, 200472 (2025). https://doi.org/10.1016/j.iswa.2024.200472
11. F. U. Khan, I. A. Shah, S. Jan, S. Ahmad, and T. Whangbo, “Machine learning-based resource management in fog computing: A systematic literature review,” *Sensors* **25**, 687 (2025). https://doi.org/10.3390/s25030687

12. S. Sheng, P. Chen, Z. Chen, L. Wu, and Y. Yao, "Deep reinforcement learning-based task scheduling in IoT edge computing," *Sensors* **21**, 1666 (2021). https://doi.org/10.3390/s21051666
13. Y. Wang and X. Yang, "Edge–cloud collaborative resource scheduling optimization based on deep reinforcement learning," in *Proc. 8th Int. Conf. on Advanced Algorithms and Control Engineering (ICAACE)* (IEEE, 2025), pp. 2065–2073. https://doi.org/10.1109/ICAACE65325.2025.11019615
14. Z. Aghapour, S. Sharifian, and H. Taheri, "Task offloading and resource allocation based on deep reinforcement learning in IoT edge environments," *Computer Networks* **223**, 109577 (2023). https://doi.org/10.1016/j.comnet.2023.109577
15. Q. Liu *et al.*, "Deep reinforcement learning for load-balancing-aware network control in IoT edge systems," *IEEE Transactions on Parallel and Distributed Systems* **33**(6), 1491–1502 (2022). https://doi.org/10.1109/TPDS.2021.3116863
16. X. Zhou *et al.*, "Edge-enabled two-stage scheduling based on deep reinforcement learning for Internet of Everything," *IEEE Internet of Things Journal* **10**(4), 3295–3304 (2023). https://doi.org/10.1109/JIOT.2022.3179231
17. J. Li, M. Dai, and Z. Su, "Energy-aware task offloading in the Internet of Things," *IEEE Wireless Communications* **27**(5), 112–117 (2020). https://doi.org/10.1109/MWC.001.1900495
18. S. Chen, Z. You, and X. Ruan, "Privacy- and energy-aware data aggregation offloading for fog-assisted IoT networks," *IEEE Access* **8**, 72424–72434 (2020). https://doi.org/10.1109/ACCESS.2020.2987749
19. C. Swain *et al.*, "METO: Matching-theory-based efficient task offloading in IoT–fog networks," *IEEE Internet of Things Journal* **8**(16), 12705–12715 (2021). https://doi.org/10.1109/JIOT.2020.3025631
20. D. Wu *et al.*, "UbiFlow: Mobility management in urban-scale software defined IoT," in *Proc. IEEE INFOCOM* (IEEE, 2015), pp. 208–216. https://doi.org/10.1109/INFOCOM.2015.7218384
21. J. M. Llopis, J. Pieczerak, and T. Janaszka, "Minimizing latency of critical traffic through SDN," in *Proc. IEEE Int. Conf. on Networking, Architecture and Storage (NAS)* (IEEE, 2016), pp. 1–6. https://doi.org/10.1109/NAS.2016.7549408
22. P. Du, P. Putra, S. Yamamoto, and A. Nakao, "A context-aware IoT architecture through software-defined data plane," in *Proc. IEEE Region 10 Symposium (TENSYMP)* (IEEE, 2016), pp. 315–320. https://doi.org/10.1109/TENCONSpring.2016.7519425
23. R. Beraldi, A. Mtibaa, and H. Alnuweiri, "Cooperative load balancing scheme for edge computing resources," in *Proc. 2nd Int. Conf. on Fog and Mobile Edge Computing (FMEC)* (IEEE, 2017), pp. 94–100. https://doi.org/10.1109/FMEC.2017.7946414
24. A. Kapsalis, P. Kasnesis, I. S. Venieris, D. I. Kaklamani, and C. Z. Patrikakis, "A cooperative fog approach for effective workload balancing," *IEEE Cloud Computing* **4**(2), 36–45 (2017). https://doi.org/10.1109/MCC.2017.25
25. L. Tong, Y. Li, and W. Gao, "A hierarchical edge cloud architecture for mobile computing," in *Proc. IEEE INFOCOM* (IEEE, 2016), pp. 1–9. https://doi.org/10.1109/INFOCOM.2016.7524340
26. J. Huang, J. Wan, B. Lv, Q. Ye, and Y. Chen, "Joint computation offloading and resource allocation for edge–cloud collaboration in Internet of Vehicles via deep reinforcement learning," *IEEE Systems Journal* **17**(2), 2500–2511 (2023). https://doi.org/10.1109/JSYST.2023.3249217
27. D. Singla, D. Gupta, and N. Goyal, "IoT-based monitoring for the growth of basil using machine learning," in *Proc. 10th Int. Conf. on Reliability, Infocom Technologies and Optimization (ICRITO)* (IEEE, 2022), pp. 1–5. https://doi.org/10.1109/ICRITO56286.2022.9964779
28. B. T. Agyeman, B. Decardi-Nelson, J. Liu, and S. L. Shah, "A semi-centralized multi-agent reinforcement learning framework for efficient irrigation scheduling," *Control Engineering Practice* **155**, 106183 (2025). https://doi.org/10.1016/j.conengprac.2024.106183
29. H. Taghavifar, C. Hu, C. Wei, A. Mohammadzadeh, and C. Zhang, "Behaviorally aware multi-agent reinforcement learning with dynamic optimization for autonomous driving," *IEEE Transactions on Automation Science and Engineering* **22**, 10672–10683 (2025). https://doi.org/10.1109/TASE.2025.3527327