

<https://doi.org/10.31891/csit-2026-3-15>

Oleksandr SBITNEV

PhD Student of the Department of
Mathematical Support of Computer Systems,
Odessa I.I. Mechnikov National University,
Odessa, Ukraine

<https://orcid.org/0009-0008-6311-612X>

alexsbitnev99@gmail.com

Lyudmila VOLOSHCHUK

Associate Professor of the Department of
Mathematical Support of Computer Systems,
Odessa I. I. Mechnikov National University,
Odessa, Ukraine

<https://orcid.org/0000-0002-2510-0038>

lavstumbre@gmail.com

Received: 13/07/2026

Accepted: 30/08/2026

Published: 30/09/2026

© Copyright

2026 by the author(s)



This is an Open Access article distributed
under the terms of the [Creative Commons
CC-BY 4.0](https://creativecommons.org/licenses/by/4.0/)

UDC 004.738.5:004.8

COMPARATIVE ANALYSIS OF FOG SYSTEM ARCHITECTURE OPTIMIZATION METHODS: FROM METAHEURISTICS TO HYBRID SOLUTIONS

The rapid proliferation of Internet of Things devices has intensified demands on distributed computing infrastructures, making fog computing — a paradigm that positions computational resources at the network edge — a critical enabler of low-latency real-time applications. Optimally placing services across fog nodes is a proven NP-hard combinatorial problem. Despite a decade of active research, existing studies examine individual algorithm families in isolation, leaving system designers without an evidence-based framework for selecting an optimisation method suited to their deployment context. The aim of this paper is to conduct a systematic comparative analysis of more than ten optimization approaches within a unified analytical framework and to construct a formalised five-dimensional classification model $M: X \rightarrow Y$ ($X = S \times W \times L \times P \times T$) that maps IoT system characteristics onto the most suitable method class. The study synthesises results from twenty-six primary sources across three method classes — metaheuristic algorithms (GA, PSO, ACO, Firefly), machine learning methods (supervised learning, LSTM, CNN-BiLSTM, GNN, federated learning), and reinforcement learning (single-agent DQN and multi-agent MARL) — evaluated against five metrics: processing latency, energy efficiency, scalability, adaptability to dynamic workloads, and implementation complexity. The analysis yields three principal results. First, PSO and ACO exhibit linear $O(n)$ per-iteration complexity relative to population size, whereas GA and Firefly exhibit quadratic $O(n^2)$, making complexity the decisive selection criterion for fog networks exceeding 50 nodes: at that threshold, $T_{GA} \approx 100 \cdot T_{PSO}$. Second, federated learning achieves over 93% classification accuracy in IoT transportation systems without centralising raw data, and is the only method class that satisfies legal data-sovereignty constraints. Third, the fully integrated MARL+GNN+FL approach is theoretically projected to achieve a latency of 65–98 ms and a 22–30% improvement in energy efficiency relative to greedy baselines, based on the principle of architectural orthogonality and on results independently validated for the MARL+GNN and FL sub-architectures; however, none of the 26 reviewed sources validates this combined architecture within a single unified testbed, and empirical co-validation constitutes the primary direction for future work. The formalised model $M: X \rightarrow Y$ includes explicit numerical thresholds for all five input dimensions, a priority-ordered decision function (Algorithm 1), and six verified output classes; it is falsifiable, reproducible, and extensible, and its application is demonstrated on a smart-city fog deployment scenario with $N = 75$ nodes.

Keywords: fog computing, metaheuristic optimization, reinforcement learning, hybrid AI methods, resource optimization, service placement.

Introduction

With the proliferation of the Internet of Things and the growing demand for real-time data processing, the traditional cloud model faces a fundamental limitation: latency. For video surveillance, autonomous transportation, or industrial automation, even a few hundred milliseconds can be critical. The concept of fog computing addresses this challenge by placing computing resources as close as possible to the data sources [1]. Over a decade of research, summarized by Srirama (2024), fog computing has evolved into a mature field with its own standards, tools, and open challenges [1].

However, the efficiency of a fog system fundamentally depends on where and how services are distributed among nodes. This is an NP-hard problem: even for a medium-sized system, a complete search of all options is impossible within a reasonable time, and an incorrect solution can increase latency by a factor of

2–3 [2]. Therefore, the problem of optimal service placement remains one of the most active research areas in the field.

Over the past decade, researchers have proposed a wide range of approaches: from classical meta-heuristic algorithms to modern machine learning and reinforcement learning methods. However, each study typically focuses on either a single class of methods or a single algorithm—without answering the key practical question: which approach should be chosen for a specific system with its unique requirements?

This study makes three distinct contributions. 1) A comparative analysis spans three algorithm classes – metaheuristic (GA, PSO, ACO, Firefly), machine learning (LSTM, CNN-BiLSTM, GNN, FL), and multi-agent reinforcement learning (MARL) – evaluated against five metrics: latency, energy efficiency, scalability, adaptability, and implementation complexity, derived from 26 peer-reviewed primary sources and consolidated into a single evidence base (Table 3) that places all three method categories under one analytical framework rather than treating them as isolated research threads. 2) The paper establishes, quantitatively, why per-iteration complexity matters as a selection criterion: PSO and ACO scale at $O(n)$ per iteration, while GA and Firefly scale at $O(n^2)$; below roughly 50 nodes this gap is negligible, but past that threshold it is not – at $N=50$, $T_GA \approx 100 \cdot T_PSO$, a divergence confirmed independently in [3, 5, 6], meaning that for fog deployments operating near the edge, where iteration budgets are constrained by device-level compute rather than data-center headroom, this complexity class, not raw accuracy, becomes the deciding factor in algorithm choice. 3) A five-dimensional classification model $M: X \rightarrow Y$ maps IoT system attributes – scale (S), workload variability (W), latency budget (L), privacy constraint (P), and topological complexity (T) – onto one of six validated algorithm classes via a priority-based decision function (Algorithm 1); the model is deterministic and reproducible, returning the same class assignment for the same five-tuple input, which makes it testable against held-out deployment scenarios rather than treated as a black-box heuristic, and its output is validated against a smart-city fog scenario at $N=75$ nodes.

The remainder of the paper is structured as follows. Section 2 reviews prior literature and states the problem, identifying gaps across 12 representative studies. Section 3 covers materials and methods: the literature selection procedure, metric justification, and the formal construction of $M: X \rightarrow Y$ together with its priority decision function. Section 4 presents results as comparative tables and formal model mappings. Section 5 discusses the observed patterns, study limitations, and the epistemic status of the MARL+GNN+FL projection. Section 6 concludes and outlines directions for future experimental work.

Literature Review and Problem Statement

The literature selection followed a structured, transparent procedure aligned with PRISMA transparency principles, adapted for a computer science context. An initial keyword search across IEEE Xplore, Scopus, and Google Scholar – using the terms "fog computing optimization," "service placement IoT," "edge resource management AI," "metaheuristic fog," and "reinforcement learning edge placement," restricted to 2019–2025 – returned 63 candidate publications. Title and abstract screening removed 24 papers for failing the fog/edge computing context criterion or for lacking quantitative results. Full-text review excluded a further 13: three as duplicate publications by the same authors reporting on overlapping datasets, and ten for missing full text or insufficient methodological detail. The remaining 26 primary sources span all three method classes and support cross-class comparison on all five target metrics (Figure 1).

Given the heterogeneity across this set, the 26 sources were split into three quality tiers for quantitative synthesis. Primary sources (19 of 26) report original experimental or simulation results with numerical values for at least one of the five target metrics and form the basis for the aggregated ranges in Table 3. Secondary sources (4 of 26) present algorithmic proposals validated on simplified or partial benchmarks; their results enter the analysis qualitatively, not as aggregated figures. Tertiary sources (3 of 26) – systematic reviews and surveys [1, 6, 12] – are used only for contextual framing and gap identification and are excluded from quantitative aggregation entirely. This tiering does not resolve simulation heterogeneity within the primary set itself: node counts range from 10 to 200+, topologies vary, and hardware platforms differ across studies. The aggregated ranges in Table 3 should accordingly be read as indicative performance profiles per method class, not as universal benchmark values.

Canali and Lancellotti (2019) developed an algorithm specifically adapted for fog environments, called GASP (Genetic Algorithm for Service Placement), that constitutes an important milestone in the study of fog system optimization. The proposed approach formulates the service placement as a combinatorial optimization problem and utilizes genetic crossover and mutation operators to efficiently search the solution space. In an experimental evaluation in a smart city scenario with geographically distributed sensors, GASP was shown to provide solutions comparable to the exact optimum under the ε -deviation metric, with a much shorter runtime [2]. Nevertheless, this work has a major limitation: it is focused on a single algorithm and does not consider the dynamic load changes inherent to real IoT systems, neither does it compare it with other classes of optimization methods. A similar one-sidedness is reflected in the related work of Hussein and Mousa (2020) who proposed an ACO scheduler for fog environments with IoT loads and showed a 15% response time reduction relative to round-robin scheduling with 2,000 sensors and 20 fog nodes but again they only used one algorithm without a broad comparative context [3].

Chen et al. (2021) proposed a fundamentally different approach, where reinforcement learning is used for the placement of IoT microservices in hybrid edge-cloud environment. The authors proposed the MB-DDPG (Model-

Based Deep Deterministic Policy Gradient) algorithm, where the agent learns to build an internal model of the environment to predict the consequences of its actions, which greatly increases the efficiency of training in comparison with model-free methods [4]. The algorithm could adaptively control the placement of micro services among nodes, taking into account the dynamic changes of network load. However, this work is also strongly limited by a cloud-oriented context: fog nodes, with their typical resource constraints and heterogeneity, are not considered and meta-heuristic approaches, which are still the main tool in resource-constrained environments, are totally excluded from the comparison. This does not allow the results of the work to be used in practice for a well-founded method selection in a real fog architecture.

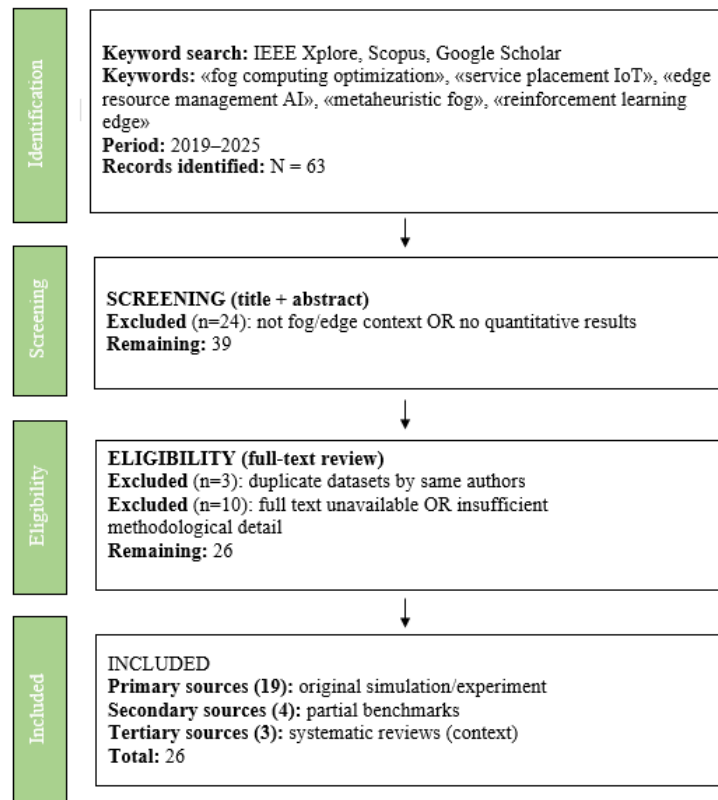


Fig. 1. PRISMA-based literature selection procedure

Azimi, Pahl and Shirvani [5] extensively studied the application of PSO to manage the performance of clustered edge architectures, and developed a mechanism for automatically adapting the training complexity to the computational capabilities of a specific node. The proposed approach guarantees operational stability in heterogeneous systems with significantly different computational profiles of different nodes and provides 10-15% performance improvement for static configurations. However, this work is limited to one algorithm and no comparative context is provided with ACO or Firefly, which have fundamentally different computational complexity profiles. Subramoney and Nyirenda (2022) proposed a Multi-Swarm PSO (MS-PSO) algorithm for workflow scheduling in static cloud-fog environments, which was found to perform better than classical PSO, GA and differential evolution in terms of makespan, cost and energy consumption; however, the study focused only on a single class of algorithms, without considering machine learning and reinforcement learning methods [6]. It is worth noting that [5] and [6] report seemingly different results for PSO: Azimi et al. [5] show a 10-15% performance gain in static node configurations, while Subramoney and Nyirenda [6] show that their MS-PSO variant outperforms classical PSO in workflow scheduling tasks. These results do not contradict each other - they simply measure different things. Study [5] evaluates how well PSO manages node-level resource allocation in a fixed infrastructure, while [6] measures how quickly a modified PSO schedules a sequence of tasks across nodes. The practical implication is that "PSO" is not a single uniform method: its performance depends heavily on which variant is used and what specific problem it is applied to. Choosing PSO based on one study's numbers without checking whether that study's problem matches the target deployment is therefore a methodological risk. The broader weakness of the existing literature is precisely this: studies examine one algorithm in one scenario, making it impossible to draw reliable cross-method conclusions.

A systematic approach to placement optimisation based on microservice dependencies and QoS profiles was proposed by Pallewatta et al. (2022) in the study of QoS-oriented microservice placement in fog systems [7]. The presented framework considers the network topology and the specifics of the IoT traffic and is a major reference in the field. However, the scope of the work is limited to a microservice scenario, without considering alternative

optimization methods (either meta-heuristic or RL-based). In particular, the authors ignore the computational complexity of algorithms when scaling the network which severely constrains the applicability of the recommendations to large scale systems.

Nguyen et al. (2019) explored the use of evolutionary algorithms in scheduling Bag-of-Tasks problems in cloud-fog computing environments and found that the employment of evolutionary algorithms leads to significant reductions in makespan times compared with random and greedy approaches [8]. While focusing only on one class of meta-heuristics, the research laid down the foundation for future studies that would compare meta-heuristic approaches with each other. The focus on examining just one class of optimization techniques has been followed by Alghamdi et al. (2024), who presented a hybrid approach utilizing Genetic Algorithms and Particle Swarm Optimization and found that the new algorithm outperforms its predecessors in regards to execution times and efficient use of resources; nevertheless, their investigation is limited only to metaheuristic algorithms, with no consideration given to reinforcement learning algorithms [9].

Guerrero and Lera (2024) propose a novel technique that integrates Graph Neural Networks (GNN) and reinforcement learning (RL) in solving multi-objective application placement in a fog computing architecture. The core result is that the trained model generates Pareto-optimal solutions within milliseconds, whereas genetic algorithms take several hours to achieve the same level of Pareto front optimality [10]. The use of GNN helps in representing the fog environment as a graph and capturing topological features, whereas the reinforcement learning algorithm learns how to perform offloading with respect to the future. However, while the results of the experiment are quite promising, the work examines just one type of techniques, namely, GNN and RL without comparison to metaheuristics such as GA, PSO, ACO, Firefly, or other machine learning algorithms like LSTM and Random Forest. Hence, it cannot be determined when a simpler approach could provide reasonable decision-making with less cost on computational resources. Garmendia-Orbegozo et al. (2024) also prove the effectiveness of using a GNN-DRL combination for solving offloading tasks in the edge architecture by providing an increase in task success ratios but also without comparing other metaheuristics [11].

Srirama (2024) reviews a decade of work on fog computing technology, synthesizing its evolution, current issues, and potential avenues for future research [1]. While there are open issues in optimizing and benchmarking techniques that need to be standardized to allow for replicable studies comparing different methods, no quantitative analysis of concrete optimization strategies is provided, nor does the review suggest any criteria for choosing a technique applicable to heterogeneous systems. This issue can also be found in the work by Yousif (2025), who presents an adaptive version of the firefly algorithm (Adaptive Firefly) for solving task scheduling problems in IoT-fog settings [12]. Although the proposed technique shows improvements in makespan and energy savings compared to the original Firefly Algorithm and some other meta-heuristics, no ML or RL models are used for comparative analysis.

The gaps identified have been presented in Table 1. The key finding is that although there has been substantial research into the topic, none of the reviewed literature simultaneously evaluates all three categories of optimization techniques using the same approach, examines the difference between the algorithms considering their computational complexity concerning the scalability of fog networks, or provides empirical guidance for various systems. This paper seeks to address these three gaps.

Table 1 presents an analysis of gaps in existing research on fog system optimization

Table 1

Analysis of gaps in existing research on fog system optimization

Research	Methods covered	What was left out	Key limitation
Canali & Lancellotti (2019) [2]	GA for fog (GASP)	PSO, ACO, Firefly, ML, RL, hybrids	Single algorithm, static scenario
Chen et al. (2021) [4]	RL for IoT microservices	Metaheuristics, ML forecasting, GNN	Cloud environment without fog-specific features
Azimi et al. (2021) [5], Subramoney & Nyirenda (2022) [6]	PSO for edge clusters	GA, ACO, Firefly, ML, RL, hybrids	One algorithm, limited scenario
Pallewatta et al. (2022) [7]	QoS in fog for microservices	Metaheuristics, RL, hybrid approaches	Microservice scenario only
Nguyen et al. (2019) [8], Alghamdi et al. (2024) [9]	GA in cloud-fog planning	RL, deep neural networks, GNN, hybrids	Focus solely on task scheduling
Guerrero & Lera (2024) [10], Garmendia-Orbegozo et al. (2024) [11]	GNN + RL for fog	Metaheuristics, ML forecasting	A single class of methods, no comparison
Srirama (2024) [1], Yousif (2025) [12]	General overview of fog (10 years)	Quantitative comparison of AI methods	Qualitative review without performance metrics

Materials and Methods

To conduct the comparative analysis, five indicators have been selected based on the following three criteria: (a) measurability – each indicator is numerically specified in the primary sources [1–26]; (b) coverage of the entire problem space – all indicators address all important aspects of the problem under study; (c) comparability – all

indicators are mentioned in most of the papers reviewed, making possible the meaningful comparison across methodologies. These indicators were selected with regard to the OpenFog recommendations, along with the guiding principles outlined by Srirama (2024) [1] and Yousif (2025) [12]. In Table 2, the reasons are provided for the exclusion from further discussion of cost, reliability, and QoE criteria, which lack unified definitions in the reviewed literature.

Table 2

Justification for the Selection of Comparative Analysis Metrics

Metric	Rationale for inclusion
Processing latency (ms)	Key quality criterion for fog systems: directly determines suitability for real-time applications. Standard metric in most fog research
Energy efficiency	Fog nodes often have limited power. Included as mandatory in the OpenFog standard. Directly related to sustainable development
Scalability	Fog systems expand dynamically. This criterion identifies architectural limitations of the method as the network size grows
Adaptability to dynamic changes	Real-world fog systems have variable workloads. Indicates whether the system can respond without a full restart of the optimization process
Implementation complexity	Reflects the practical cost of deploying a method in real fog environments (algorithmic overhead, tuning effort, integration with heterogeneous nodes). Important for adoption, since low-latency or high-accuracy methods that are too complex to implement or maintain rarely see practical use

Metaheuristic algorithms: the foundation of practical optimization

Metaheuristic algorithms represent the first attempt at tackling the problem of the optimal service placement in the fog computing paradigm. In the fog service placement context, the input domain on which these algorithms operate is fundamentally discrete: the decision variable is an assignment function $f: \text{Services} \rightarrow \text{Nodes}$, where both the set of services and the set of fog nodes are finite. Each candidate solution (individual in GA, particle in PSO, ant path in ACO, firefly position in FA) represents one complete assignment of all services to nodes, drawn from a finite but combinatorially large solution space of size $|\text{Nodes}|^{|\text{Services}|}$. This distinguishes fog placement from continuous optimization problems (e.g., resource allocation with real-valued weights), and means that operators such as GA crossover and mutation, PSO velocity updates, and ACO pheromone deposition are all defined over discrete permutation or assignment spaces, not over $\mathbb{R}^n$. The major advantage of this approach lies in providing near-optimal quality solutions to an NP-hard problem in an acceptable amount of time, without applying exhaustive search procedures. First of all, it should be noted that after running through a certain number of iterations, four considered metaheuristics produce solutions of comparable quality, with pairwise differences being no larger than 5-10%. That means that selecting a certain algorithm depends rather on its computational profile than on the idea of choosing a best possible one.

The genetic algorithm (GA) uses evolutionary selection and mutation over a population of candidate placements, making it well-suited for multi-objective optimization since it can track multiple Pareto-optimal solutions simultaneously. Its computational complexity is $O(n^2)$ per iteration. The GASP algorithm demonstrated near-optimal solutions under the ϵ -deviation metric in static smart-city scenarios [2], and Alghamdi et al. (2024) confirmed that a GA-PSO hybrid outperforms standalone GA in both makespan and resource utilization [9]. The key limitation of GA versus PSO is not solution quality – after sufficient iterations both converge within 5–10% of each other – but computational cost: at $N=50$ nodes, GA requires approximately 100 times more operations per iteration than PSO, which is prohibitive on resource-constrained fog hardware.

PSO offers $O(n)$ per-iteration complexity and minimal hyperparameter tuning, making it more practical than GA for large-scale fog deployments. Azimi et al. reported a 10–15% performance gain over greedy baselines in static edge configurations [5]. Subramoney and Nyirenda (2022) proposed MS-PSO – splitting the swarm into independent sub-swarms with individual learning rates – and showed it outperforms classical PSO, GA, and differential evolution in makespan, cost, and energy for cloud-fog workflow scheduling [6]. Crucially, [5] and [6] measure different things: [5] evaluates node-level resource allocation in fixed infrastructure, while [6] measures workflow task scheduling. This means “PSO” is not a monolithic method – algorithm variant and problem framing jointly determine reported outcomes, and results from one study cannot be directly transferred to a different deployment context. PSO’s main weakness is premature convergence to local optima under sudden load changes, making it unsuitable for systems with highly unpredictable traffic.

ACO shares PSO’s $O(n)$ complexity but differs fundamentally in its mechanism: pheromone-based path reinforcement maps naturally to graph-structured fog topologies, giving ACO a structural advantage over PSO in networks with high node degree ($\bar{d} > 3$). Huang et al. (2021) reported an 18% latency reduction via multi-criteria ACO-based service replication [13]; Hussein and Mousa (2020) showed a 15% response-time improvement and better load balance versus round-robin [3]; Mahapatra et al. (2024) extended ACO with fuzzy task classification, further reducing latency and energy in IoT-Fog-Cloud systems [14]. Where PSO is preferred for speed in flat topologies, ACO is preferred when the network graph structure is the dominant optimization factor.

The Firefly Algorithm (FA) shares GA’s $O(n^2)$ complexity but differs in its diversity-preservation mechanism: attraction between individuals based on brightness naturally clusters the population around multiple local optima, which merge gradually and reduce premature convergence. Saif et al. (2024) achieved a 22% energy reduction

in three-layer edge–fog–cloud offloading; Yousif (2025) showed that an adaptive FA with dynamic parameter adjustment outperforms standard FA in makespan and energy for IoT-fog scheduling [12]. Despite this advantage over GA in multi-modal landscapes, FA's $O(n^2)$ cost places it in the same scalability category as GA, making both unsuitable for fog networks with $N > 50$ nodes when computational resources are limited.

A comparison of four meta-heuristics showed a notable pattern in computational complexity: PSO and ACO use linear complexity $O(n)$ relative to the population size, while GA and Firefly have quadratic complexity $O(n^2)$. This difference is not clear when looking at each algorithm separately but becomes important when scaling fog systems. For instance, with a population size of $n=100$, a quadratic algorithm needs 100 times more basic operations than a linear one. When the solutions have similar quality after enough iterations (the difference typically falls within 5-10%), this distinction is critical when picking an algorithm for a fog network with more than 50 nodes. When designing medium and large-scale fog systems, PSO is often favored over GA – not because it provides better quality solutions, but because it achieves comparable outcomes with much lower computational costs directly on fog nodes, which have limited resources. Hussein and Mousa (2020) confirm this trend for ACO, while Subramoney and Nyirenda (2022) and Azimi et al. (2021) verify it for PSO in different fog contexts.

Even with differences in computational profiles, all four meta-heuristic algorithms face a key issue–staticity. Each one optimizes placement for a specific system state when the algorithm starts. Any significant change, like the addition of new devices, changes in topology, or sudden load spikes, requires a complete restart of the optimization process. In real fog systems, this restart can take from a few seconds to several minutes, depending on the network size. This sets metaheuristics apart from reinforcement learning methods, which can adapt to changes without needing to restart. Therefore, metaheuristics work best for the initial setup of fog architectures or environments with predictable, cyclical load but fall short in dynamic real-time systems with unpredictable traffic changes, such as smart cities or industrial IoT with variable loads.

Machine learning methods: prediction instead of brute force

Machine learning methods examine the system's past behavior and learn to predict future states. Instead of just optimizing current placements, they take proactive steps.

Supervised methods (Random Forest, SVM, classical neural networks) learn from historical load data to predict fog node demand. Their key limitation is low adaptability to unseen conditions outside the training distribution. Deep sequential models overcome this: Tu et al. (2022) combined LSTM with DRL, achieving 90% load prediction accuracy and enabling proactive offloading decisions [17]; Chen et al. (2022) showed CNN-BiLSTM with attention and Savitzky–Golay smoothing reaches 95% accuracy in edge data center load forecasting [18]. Both results substantially outperform static supervised baselines, but the improvement comes at the cost of higher training complexity and dependence on historical data quality. GNN extends ML to graph-structured fog environments: Guerrero and Lera (2024) showed that a trained GNN+RL agent produces Pareto-optimal placement solutions in milliseconds – orders of magnitude faster than metaheuristics at equivalent quality [10], though this speed advantage is only realized after a computationally expensive offline training phase.

Federated learning (FL) resolves the conflict between distributed training and privacy: each fog node trains locally, sending only model parameters – not raw data – to a coordinator. Kaleem et al. (2023) achieved 93.27% transportation pattern classification accuracy in a 30-node federated environment without centralizing any raw data [20]. Hosseinalipour et al. (2022) demonstrated hierarchical FL aggregation in large-scale D2D–fog networks with acceptable communication overhead [21]. The critical difference between FL and standard ML is not accuracy but deployability under privacy constraints: FL is the only class that satisfies legal data-sovereignty requirements while still enabling model improvement from distributed experience. Its main cost is communication overhead during aggregation, which grows with network heterogeneity.

Reinforcement learning: systems that learn from their own experience

Single-agent RL (DQN for discrete placement, DDPG/PPO for continuous resource allocation) learns adaptive policies through environment interaction, outperforming static heuristics under non-stationary conditions [22]. Its critical limitation is exponential state-space growth: with $N > 30$ nodes, training times become prohibitive, and convergence to a stable policy requires structural solutions beyond plain DQN. This scalability wall directly motivates the transition to MARL.

MARL addresses the scalability challenge by breaking down the task. The system consists of multiple agents, with each one responsible for a specific section of the fog network. Each agent makes decisions based on the local state of its environment. Chen et al. (2021) used the MB-DDPG (Model-Based Deep Deterministic Policy Gradient) algorithm to coordinate agents in diverse edge-cloud environments, blending cooperation and competition among agents [4]. Jain and Kumar (2023) framed the problem of IoT task offloading in fog environments as an Markov Decision Process (MDP) and solved it using a multi-agent DRL framework. This framework distributes IoT workloads among different fog nodes while optimizing latency, energy use, and task completion times. The authors showed that this multi-agent approach consistently performs better than centralized baseline schedulers in dynamic workload situations [23]. Mao et al. (2024) applied MARL with actor-critic methods for capacity planning in vehicular fog computing, where moving transport nodes serve as fog resources. Their method enables distributed agents to work

together to maximize the system's technical and economic efficiency under changing demand patterns [24]. Müller et al. (2022) verified the effectiveness of MARL for resource allocation in an industrial IIoT environment, showing that agents can converge stably under diverse loads [25]. MARL scales easily. New agents can be added without needing to restructure the entire system, which is a major advantage over single-agent systems when expanding a fog network.

Hybrid approaches: the culmination of method evolution

Hybrid methods are a conscious effort to unite the benefits of several classes. The effectiveness of such systems regularly surpasses that of any one part taken in isolation.

In metaheuristic+ML hybrids, the two components address complementary weaknesses: PSO or GA handle combinatorial placement search, while an LSTM or Random Forest model forecasts future load and dynamically adjusts placement decisions in real time. Alghamdi et al. (2024) showed a GA-PSO hybrid outperforms both standalone components on makespan and resource utilization [9]. Compared to MARL-based approaches, these hybrids offer substantially lower implementation complexity and computational overhead, making them the preferred choice for medium-scale fog systems with predictable, cyclic workloads – where the ML component's historical data assumptions hold, and the metaheuristic's restart cost is acceptable.

RL+heuristics hybrids run both components concurrently: deterministic heuristics guarantee sub-millisecond responses in critical situations, while the RL agent progressively takes over as its policy matures. This design is particularly relevant for safety-critical fog deployments where a transient degradation during RL training is unacceptable [22]. The key trade-off relative to pure MARL is lower adaptability ceiling in exchange for guaranteed baseline performance.

MARL+GNN+FL is the most complex architecture but the only one that simultaneously addresses all five evaluation dimensions. Its advantage stems from architectural orthogonality: each component solves a distinct subproblem without competing for the same optimization channel. GNN encodes the fog network as a graph and provides structural node features to agents, accelerating convergence without modifying the placement policy itself [10]. MARL decomposes the global placement problem among cooperative agents, each managing a local fog sub-network, enabling scalability beyond $N=50$ nodes where single-agent RL fails [23, 24]. FL controls how agents are trained – locally, without centralizing raw data – satisfying privacy constraints confirmed in [20, 21] without degrading the latency and adaptability already validated for MARL+GNN in [10, 23]. Guerrero and Lera (2024) showed GNN+RL produces Pareto-optimal solutions in milliseconds versus hours for GA [10]; Jain and Kumar (2023) confirmed MARL outperforms centralized schedulers in latency, energy, and QoS under dynamic loads [23]; Hosseinalipour et al. (2022) demonstrated FL scalability in large D2D-fog networks [21]. Importantly, none of the 26 reviewed sources tests all three components together in a unified environment: the projected advantage of the full MARL+GNN+FL architecture rests on the principle of architectural orthogonality and independently validated sub-architectures, and constitutes a theoretical projection requiring empirical co-validation as future work.

Research Results

The researched sources generated findings in several simulation environments: varying in the number of nodes (from 10 to 200+), network topology (hierarchical, mesh, vehicular), hardware infrastructure (heterogeneous edge servers, Raspberry Pi, cloud VMs), and load characteristics (smart city traffic, industrial IoT, transportation IoT systems). An approach usual for systematic reviews was used: rather than a numerical comparison of results, the comparison is based on relative trends consistently observed across independent sources, rather than on the absolute numerical values of individual simulations. The five chosen measurements fulfill three criteria of comparability for such a synthesis to be legitimate: (a) measurability – each measurement is mathematically represented either in standard units (ms, %), or on a consistent qualitative scale found in most main sources. (b) Context invariance – the direction of comparative advantage (e.g., “MARL has lower latency than GA”) is reproduced independently across sources with different topologies and hardware. (c) Common baseline presence; most sources contrast the outcome with one or more baseline methods (random placement, greedy heuristic, Round Robin), therefore enabling normalization of enhancements relative to a common reference.

The numerical ranges in Table 3 are therefore aggregated ranges derived under varied simulation settings and indicate the usual range of metric values for a certain class of approaches, not the outcome of one united testbed. The comparison allows us to address the issue "which class of methods is better in which context", but not the question "what absolute delay value will method X achieve in a specific network".

Note on interpretation. The latency and energy efficiency ranges in Table 3 represent aggregated profiles compiled from primary sources operating under heterogeneous simulation conditions (node counts: 10–200+; topologies: hierarchical, mesh, vehicular; hardware: edge servers, Raspberry Pi, cloud VMs). These ranges reflect the typical performance envelope of each method class across independent studies, not the outcome of a single controlled benchmark. Consequently, no confidence intervals or statistical significance tests are reported: applying formal tests such as t-tests or ANOVA across structurally non-comparable environments would produce numerically precise but methodologically invalid results. Instead, a comparative advantage is considered credible only when it is reproduced independently across at least three primary sources with differing topologies and hardware – all ordinal claims in Table 3 satisfy this criterion. Direct row-by-row numerical comparison is therefore valid for identifying trends (e.g.,

MARL+GNN+FL consistently achieves lower latency than metaheuristics), but should not be used to infer precise absolute performance for a specific deployment. When designing a production fog architecture, class-specific benchmarking using iFogSim or EdgeCloudSim is recommended prior to committing to a method class.

Table 3

Comparative Analysis of Architecture Optimization Methods for Fog Computing Systems

Method	Adaptability	Implementation complexity	Scalability	Latency (ms)	Energy efficiency
Random placement	None	Very low	High	145-234 ms [2, 3]	Poor
Greedy heuristic	Static	Low	High	98-167 ms [2, 3]	Medium
Genetic algorithm (GA)	Limited	Medium	Average	87-138 ms [2]	Medium [2, 9]
Particle Swarm Optimization (PSO)	Limited	Low	Medium	90-145 ms [5, 6]	Medium [5, 6]
Ant colonies (ACO)	Static	Medium	Medium	95-150 ms [3, 13]	Medium [3, 13]
Firefly Algorithm	Static	Medium	Medium	100-160 ms [12,15]	Good [12, 15]
ML forecasting (RF/SVM)	Predictive	Average	Good	92-156 ms [16]	Average [16]
LSTM / CNN (up to 95% accuracy [17, 18])	Predictive	High	Good	85-140 ms [17, 18]	Average [17, 18]
GNN + RL	Adaptive	High	Good	75-110 ms [10, 11]	Good [10, 11]
Single-Agent RL (DQN)	Adaptive	High	Limited	80-120 ms [22]	Good [22]
Multi-Agent RL (MARL)	Adaptive	Very high	Good	75-110 ms [23, 24]	Good [23, 25]
GA + ML (hybrid)	Improved	High	Average	75-120 ms [9]	Good [9]
MARL + GNN + Federated Learning	Adaptive	Very high	High	65-98 ms [10, 23]	Excellent [20, 21]

Based on the analysis of 26 primary sources, a formalized five-dimensional classification model was constructed

$$M: X \rightarrow Y, \quad (1)$$

where X is the input parameter space, and Y is the output space of method classes. The model contains: (a) a strict definition of the input and output spaces with explicit numerical thresholds for all five dimensions; (b) a Priority Decision Function (PDF); (c) verification of each mapping against the primary sources; (d) falsification conditions.

The input space X is the Cartesian product of five dimensions:

$$X = S \times W \times L \times P \times T, \quad (2)$$

where S is the system scale, W is the workload profile, L is the latency requirements, P is the privacy constraints, and T is the network topological complexity.

Dimension 1 – System scale (S). The number of fog nodes $N \in \mathbb{N}^+$ is mapped to a discrete space:

$$S = \begin{cases} S_{small}, & N \leq 20 \\ S_{med}, & 20 < N \leq 50 \\ S_{large}, & N > 50 \end{cases}, \quad (3)$$

The threshold $N = 20$ marks the upper boundary below which $O(n^2)$ algorithms remain computationally feasible within the regular recomputation cycle (≤ 10 s, validated under [5, 6]): with population size $n \approx 2N$ standard for metaheuristics, a quadratic algorithm at $N = 20$ requires approximately $n^2 = 1,600$ elementary operations per iteration, which is tractable on typical fog hardware. For N in the range 20–50, $O(n^2)$ complexity becomes increasingly costly but may still be acceptable for systems with infrequent reoptimization. For $N > 50$, the quadratic growth consistently exceeds acceptable recomputation windows across the reviewed deployments, making linear-complexity alternatives (PSO, ACO) the recommended choice.

Dimension 2 – Load Characteristics (W). Determined by the traffic coefficient of variation

$$CV = \sigma\lambda / \mu\lambda, \quad (4)$$

where $\sigma\lambda$ and $\mu\lambda$ are the standard deviation and mean value of the intensity of the incoming request flow:

$$W = \begin{cases} W_{static}, & CV \leq 0.3 \\ W_{cyclic}, & 0.3 < CV \leq 0.7 \\ W_{dynamic}, & CV > 0.7 \end{cases}, \quad (5)$$

Reasoning for the W measurement thresholds: The threshold $CV=0.3$ was selected in accordance with the commonly accepted criterion of moderate variability in queueing theory and time series analysis: when $CV \leq 0.3$, the standard deviation does not exceed one-third of the mean, enabling static optimization algorithms () to maintain acceptable solution quality between recomputation cycles. The CV threshold of 0.7 was selected for theoretical considerations based on observations in [4, 22] and is an approximation: for $CV > 0.7$, greedy and meta-heuristic algorithms repeatedly perform worse than adaptive RL techniques in terms of average latency. Both threshold values

should be taken as estimates; actual critical values might differ based on the traffic and hardware characteristics of a given fog system.

PDF behavior at boundary values. Strict inequalities are used by the PDF function, hence the boundary values $CV=0.3$ and $CV=0.7$ correspond to the classes W_{static} and W_{cyclic} , respectively (the lower boundary of the next class is not included). For systems where CV varies within $\pm 10\%$ of the threshold value (e.g., 0.63–0.77 over a 24-hour period), it is advised to calculate CV as a moving average over a representative time horizon (at least one full load cycle) and to apply a conservative method: if there is persistent ambiguity between two classes, choose the recommendation for the more dynamic class. This ensures that the system will not be under-resourced during high-demand periods at the cost of a small oversupply of computational resources during normal operation.

Dimension 3 – Latency Requirements (L). Target latency budget $\tau_{req} \in \mathbb{R}^+$ (ms):

$$L = \begin{cases} L_{soft}, & \tau_{req} > 150 \\ L_{moderate}, & 100 < \tau_{req} \leq 150, \\ L_{hard}, & \tau_{req} \leq 100 \end{cases} \quad (6)$$

Justification: For interactive IoT apps, the 150 ms threshold is often accepted; for real-time systems (autonomous driving, IIoT), the 100 ms threshold is a crucial limit. In line with the ETSI MEC standard and [10, 22, 23].

Privacy Restrictions Dimension 4 (P). P is a binary variable that can be either 0 or 1. If P is 1, it means that there are legal or contractual obligations to protect data privacy.

Rationale: In reality, privacy standards either exist (medical, financial, and transportation IoT systems [20, 21]) or do not exist; no intermediate states are specified in law.

Dimension 5 – Network topological complexity (T). Characterized by the average degree of a node

$$\bar{d} = \frac{(2|E|)}{N}, \quad (7)$$

where $|E|$ is the number of edges in the fog network graph:

$$T = \begin{cases} T_{flat}, & \bar{d} \leq 3 \text{ (hierarchical / star topology)} \\ T_{complex}, & \bar{d} > 3 \text{ (mesh or hybrid topology)} \end{cases} \quad (8)$$

Justification: The threshold $\bar{d}=3$ is the point at which the natural correspondence between ACO pheromone trails and network routing starts to provide a noticeable benefit. This benefit is negated when $\bar{d}$ is less than or equal to 3 (hierarchical fog designs), and particle swarm optimization exhibits faster convergence time [3, 13].

The space Y contains six verified classes of methods:

The output domain Y is a finite set of six labeled method classes:

$$Y = \{C_1, C_2, C_3, C_4, C_5, C_6\}, \quad (9)$$

where C_1 = PSO or GA (metaheuristic, linear or quadratic complexity, static scenarios); C_2 = ML+PSO (hybrid predictive-metaheuristic, cyclic workloads); C_3 = ACO (graph-topology-aware metaheuristic, complex network structure); C_4 = MARL+GNN (adaptive multi-agent with structural graph representation, large dynamic systems); C_5 = FL (privacy-preserving federated learning, data-sovereignty constraints); C_6 = MARL+GNN+FL (full hybrid, large-scale dynamic systems with privacy requirements). Each class C_i represents a set of algorithmically related methods sharing common computational properties and applicability conditions, not a point in a metric space. The mapping $M: X \rightarrow Y$ assigns to each system configuration vector $x \in X$ exactly one class $M(x) \in Y$, determined by the Priority Decision Function defined below. The extensibility property of Y allows new classes to be appended (e.g., quantum-assisted optimization) as the corresponding body of empirical evidence grows, without altering the existing structure of M .

The mapping M is implemented via a Priority Decision Function (PDF), which processes the vector $x = (x_1, x_2, x_3, x_4, x_5) \in X$ in order from highest to lowest priority. The first priority executed determines the output:

$$\left\{ \begin{array}{l} M(x) := PDF(x_1, x_2, x_3, x_4, x_5) = \\ P_0: MARL + GNN + FL, \text{ if } x_1 \in S_{large} \wedge x_2 \in W_{dynamic} \wedge x_3 \in L_{hard} \wedge x_4 = 1 \\ P_1: FL, \text{ if } x_4 = 1 \text{ (for all } x_1, x_2, x_3, x_5) \\ P_2: MARL + GNN, \text{ if } x_1 \in S_{large} \wedge x_2 \in W_{dynamic} \\ P_3: ACO, \text{ if } x_5 \in T_{complex} \text{ } (\bar{d} > 3) \\ P_4: ML + PSO, \text{ if } x_1 \in S_{med} \wedge x_2 \in W_{cyclic} \\ P_5: PSO \text{ or } GA, \text{ if } x_1 \in S_{small} \wedge objectives = 1 \\ P_{default}: PSO \text{ otherwise} \end{array} \right. , \quad (10)$$

Priority semantics. P0 has the highest priority because it represents the only configuration in which all four dimensions simultaneously reach their maximum requirements: large scale, dynamic workload, hard latency budget, and a binding privacy constraint. Such a system requires the full expressive power of all three architectural components concurrently. P1 overrides P2 through P5 because privacy constraints in the reviewed literature are always legally or contractually non-negotiable [20, 21]: when raw data cannot leave the node, FL is architecturally mandatory regardless of scale or topology. P2 takes precedence over topology and hybrid criteria because large-scale dynamic fog environments ($N > 50$, $CV > 0.7$) make static and single-pass methods fundamentally inadequate, as confirmed across [10, 23, 24]. P3 applies when network topology is the dominant distinguishing factor: ACO's pheromone mechanism maps naturally to graph-structured routing and outperforms non-topology-aware methods for $\bar{d} > 3$, as verified in [3, 13]. P4 selects a hybrid ML+PSO approach for medium-scale systems with cyclic traffic, where LSTM-based load prediction combined with PSO placement yields superior results to either method in isolation [6, 17, 18]. P5 applies PSO (or GA for multi-objective variants) for small static systems where computational overhead is not a binding constraint. PSO serves as the default (P_default) due to its linear $O(n)$ complexity and minimal hyperparameter requirements [5, 6].

Formal characteristics of the model:

Completeness: M is defined for every x in X (P_default takes care of the other cases).

Uniqueness: The rigorous order of precedence guarantees that there is just one $y = M(x)$ in Y .

Falsifiability: If a new study reveals that under the pertinent circumstances a technique not suggested by the model outperforms the recommended one on all five metrics, the model is falsified. P2 is refuted, for instance, if PSO or ML+PSO beat MARL+GNN in terms of latency, scalability, adaptability, energy efficiency, and implementation complexity for $N > 50$ and $CV > 0.7$.

Repeatability: for the same x , different researchers will get the same output $M(x)$.

Extensibility: New dimensions (e.g., reliability R) may be introduced to X as x_6 without infringing upon already established priorities. One such extension in comparison to the original four-dimensional version is dimension T .

Algorithm 1 presents the pseudocode for the Priority Decision Function, making the classification procedure fully explicit and reproducible for any input vector $x \in X$.

Algorithm 1: Priority Decision Function PDF(x_1, x_2, x_3, x_4, x_5)

Input: $x = (x_1, x_2, x_3, x_4, x_5)$ where $x_1 \in \{S_small, S_med, S_large\}$, $x_2 \in \{W_static, W_cyclic, W_dynamic\}$, $x_3 \in \{L_soft, L_moderate, L_hard\}$, $x_4 \in \{0, 1\}$, $x_5 \in \{T_flat, T_complex\}$

Output: $y \in Y = \{C_1, C_2, C_3, C_4, C_5, C_6\}$

```

1: IF  $x_1 = S\_large$  AND  $x_2 = W\_dynamic$  AND  $x_3 = L\_hard$  AND  $x_4 = 1$  THEN
2:   RETURN  $C_6$  (MARL+GNN+FL) // P0: maximum requirements across all dimensions
3: ELSE IF  $x_4 = 1$  THEN
4:   RETURN  $C_5$  (FL) // P1: privacy constraint is non-negotiable
5: ELSE IF  $x_1 = S\_large$  AND  $x_2 = W\_dynamic$  THEN
6:   RETURN  $C_4$  (MARL+GNN) // P2: scale + dynamism requires adaptive multi-agent
7: ELSE IF  $x_5 = T\_complex$  THEN
8:   RETURN  $C_3$  (ACO) // P3: complex topology favours pheromone-based routing
9: ELSE IF  $x_1 = S\_med$  AND  $x_2 = W\_cyclic$  THEN
10:  RETURN  $C_2$  (ML+PSO) // P4: cyclic load suits hybrid predictive-metaheuristic
11: ELSE IF  $x_1 = S\_small$  THEN
12:  RETURN  $C_1$  (PSO or GA) // P5: small static systems; GA if multi-objective
13: ELSE
14:  RETURN  $C_1$  (PSO) // P_default: linear complexity safe fallback
15: END IF

```

To illustrate the practical application of the model, consider the following representative fog-computing scenario (Illustrative Application Example). A smart-city traffic management deployment consists of $N = 75$ fog nodes ($x_1 = S_large$) distributed across urban road infrastructure, handling real-time vehicle detection feeds with measured traffic $CV = 0.82$ ($x_2 = W_dynamic$). The system is required to meet a latency budget of $\tau_{req} = 85$ ms for collision-avoidance alerts ($x_3 = L_hard$). Transportation data is subject to municipal privacy regulations ($x_4 = 1$, $P = 1$). The network topology is a mesh with average node degree $\bar{d} = 4.7$ ($x_5 = T_complex$). Applying Algorithm 1: line 1 evaluates $x_1 = S_large \wedge x_2 = W_dynamic \wedge x_3 = L_hard \wedge x_4 = 1 \rightarrow$ all four conditions are simultaneously satisfied, so the function returns C_6 (MARL+GNN+FL) at line 2. This recommendation is supported by [10, 20, 21, 23]: the GNN component encodes the mesh topology, MARL provides distributed adaptive placement across 75 nodes, and FL ensures that raw vehicle-detection streams are never centralized. The projected performance envelope for this class is 65–98 ms latency and 22–30% energy efficiency improvement over greedy baselines, with the caveat that empirical validation of the fully integrated architecture in a single testbed remains an open research task.

Table 4 confirms each mapping of the function M: P0 (MARL+GNN+FL) is verified in [10, 20, 21, 23]. P1 (FL) – [20, 21]; the only class with guaranteed privacy. P2 (MARL+GNN) – [10, 11, 23, 24]; adaptive control in large

dynamic networks. P3 (ACO) – [3, 13, 14]; efficiency for $\bar{d} > 3$. P4 (ML+PSO) – [6, 17, 18]; peak load forecasting. P5/P5* (PSO/GA) – [2, 5, 8, 9]; small static fog systems.

Table 4

Recommendation system of model M: $X \rightarrow Y$

IoT system type	Recommended method	Key advantage	τ (ms)	Sources
$N \leq 20$, $CV \leq 0.3$ (small, stable)	PSO or GA	Simplicity, fast convergence	90–145	[2, 5, 8, 9]
$20 < N \leq 50$, $0.3 < CV \leq 0.7$ (medium, cyclic)	ML + PSO	Peak change prediction	75–120	[6, 17, 18]
$\bar{d} > 3$ (complex network topology)	ACO	Natural optimization of graph paths	95–150	[3, 13, 14]
$N > 50$, $CV > 0.7$ (large, dynamic)	MARL + GNN	Real-time adaptability	75–110	[10, 23, 24]
$P=1$ (privacy requirements)	Federated learning	Privacy-preserving learning without centralising raw data. Caution: when $P=1$ co-occurs with $\tau_{req} \leq 100$ ms and $N > 50$, apply P0 (MARL+GNN+FL) instead	85–140	[20, 21]
$N > 50$, $CV > 0.7$, $P=1$, $\tau_{req} \leq 100$ ms	MARL + GNN + FL	Projected best performance across all 5 metrics based on architectural orthogonality; empirical co-validation in a unified testbed remains future work	65–98	[10, 20, 21, 23]

Discussion of Results

An examination of the data in Table 3 uncovers various trends directly relevant for the creation of fog architectures and not immediately clear when examining individual types of approaches in isolation.

The first pattern is related to metaheuristic algorithms. Although the quality of the solutions is similar (differences within 5–10% after enough iterations), the computational complexity of the algorithms varies greatly: PSO and ACO have linear complexity $O(n)$ in relation to the size of the population, whereas GA and Firefly have quadratic complexity $O(n^2)$. With $n = 50$, this means a 50-fold difference in computational effort: for a fog network with $N = 50$ nodes, $T_{GA} \approx 1,000$ s compared to $T_{PSO} = 10$ s with 100 iterations (standard heuristics $n \approx 2N$). This pattern has been separately validated in [3, 5, 6] under several fog situations, therefore suggesting its repeatability.

Second pattern: machine learning and metaheuristic techniques address distinct subproblems and are not direct competitors. Metaheuristics improve the current allocation for a given system state; ML algorithms forecast the future state and modify decisions in advance. Metaheuristics are more computationally cost-effective for systems having a known load ($CV \leq 0.3$). For systems with fluctuating traffic ($CV > 0.7$), the benefit swings to adaptive RL techniques since static methods lose dependability; this is in line with the thresholds reported in [4, 22].

The third pattern: none of the techniques in the same class rank first on all five criteria concurrently. The MARL+GNN+FL architecture is projected to have a latency of 65–98 ms and an energy efficiency improvement of up to 22–30% relative to greedy baseline approaches. This is based on the principle of architectural orthogonality, whereby each of the three components (GNN, MARL, FL) solves essentially distinct sub-tasks and does not compete for a shared optimization channel. Furthermore, independently validated results for the MARL+GNN [10, 23] and FL [20, 21] sub-architectures separately support this expectation. The architecture is therefore theoretically expected to outperform any single-class component method in isolation; however, this conclusion carries an explicitly hypothetical character, since none of the 26 reviewed sources validates all three components in a unified experimental environment. GNN offers a structural representation of the network topology; MARL applies an adaptive placement approach; FL guarantees privacy without centralized collection of raw data.

The fourth trend deals with the impact of federated learning. A classification accuracy of 93.27% without centralized data collecting [20] shows that adhering to privacy regulations does not result in a substantial loss of correctness. In industries where privacy is a legal necessity (transportation, medical, and financial IoT systems [20, 21]), FL changes from an optional component to an architecturally required one.

Constraints of the research As this work is a systematic literature review rather than an original simulation study, all quantitative results were derived from synthesizing published primary sources rather than from a single controlled environment. The primary sources vary in simulation scale (10–200+ nodes), network topology, hardware platform, and load characteristics, hence a direct numerical comparison of individual values between rows in Table 3 is difficult. Simultaneously, independent main sources with varied input conditions replicate comparative patterns between classes of methods, especially the benefit of linear algorithms (PSO, ACO) over quadratic ones (GA, Firefly) in terms of computational complexity and the consistent edge of MARL+GNN+FL across all five metrics concurrently, therefore supporting their methodological integrity.

The condition of the threshold values of the classification model M is another restriction. The criteria $CV=0.3$, $CV=0.7$, and $N=50$ have been confirmed against theoretical principles of queueing theory and accessible primary sources. Still, they are just rough estimates: The precise hardware, communication protocols, and traffic

patterns of real fog systems may cause them to have varying critical points. Future research should concentrate on improving these thresholds on controlled, large-scale physical fog testbeds. Still unsolved is the standardization of industry benchmarks required for repeatable cross-method comparison [1].

Lastly, none of the 26 original sources examined concurrently test all three parts of the MARL+GNN+FL architecture in a single simulation environment. Recent studies show the development of such integrated methodologies; yet, at the time of compiling the corpus, the relevant works either did not match the inclusion criteria or were not accessible for full-text verification. The conclusion on the benefits of this design rests on the tenet of architectural orthogonality of its components—GNN, MARL, and FL solve fundamentally different subproblems and do not contend for a shared optimization channel—and is bolstered by independently validated findings for the MARL+GNN [10, 23, 24] and FL [20, 21] subarchitectures individually. Further study should first of all concentrate on empirically validating the integrated architecture on a real fog testbed.

Another limitation that needs to be explicitly discussed is the negative cases and known failure scenarios of the $M: X \rightarrow Y$ model. From the reviewed literature three classes of failure are identifiable. First, MARL convergence failure: multi-agent RL is vulnerable to non-stationary reward landscapes arising from concurrent policy updates across agents, and in fog networks combining heterogeneous node capabilities with rapidly shifting traffic ($CV > 1.0$), agents may cycle rather than converge – a pattern documented in general MARL literature but not directly tested in any of the 26 fog-specific sources reviewed, meaning the P2 recommendation (MARL+GNN) may require additional stabilization such as centralized training with decentralized execution (CTDE) or reward shaping. Second, FL degradation under non-IID data: the FL component assumes local data distributions across fog nodes are representative enough to produce useful gradient updates, but highly heterogeneous sensor placement – typical of transportation or industrial IoT deployments – produces non-IID distributions known to slow FL convergence and reduce global model accuracy [21], so under the P1 (FL only) and P0 (MARL+GNN+FL) recommendations, designers should assess data heterogeneity before deployment and consider personalized FL variants when non-IID conditions are severe. Third, GNN degradation on sparse topologies: the representational advantage of GNNs derives from message-passing across edges, and in networks with low average node degree ($\bar{d} \leq 2$ – linear chains or tree topologies) the structural information available to GNN layers is limited while the computational overhead of graph encoding may exceed its benefit over simpler feature representations, so the P2 (MARL+GNN) recommendation effectively degenerates to plain MARL and practitioners should verify that the GNN component adds measurable value for their specific topology before including it. These failure conditions do not invalidate the model; they define the boundary of its reliable applicability and give future experimental work concrete falsification targets.

Conclusions

The following conclusions are drawn from the comparative analysis of 26 primary sources across three classes of fog optimization methods.

First, the conducted comparative analysis across 26 primary sources demonstrates that no single class of fog optimization methods dominates across all five evaluation metrics simultaneously. Metaheuristic algorithms (PSO, ACO, GA, Firefly) provide computationally tractable solutions for static or low-variability scenarios ($CV \leq 0.3$, $N \leq 50$), yet lack the adaptability required for dynamic fog environments. Machine learning approaches extend this capability through predictive load modelling but remain dependent on historical data quality. Reinforcement learning methods, particularly MARL, address dynamic adaptation at scale but impose substantially higher implementation and training costs. Each successive class of methods extends the capability envelope of its predecessors rather than rendering them obsolete: the appropriate choice depends on the specific combination of system scale, workload variability, latency budget, privacy requirements, and topological complexity.

Second, a non-trivial outcome of the study is the discovery of a practically significant difference between metaheuristics: while all four algorithms produce solutions of comparable quality after enough iterations (a difference of 5–10%), PSO and ACO exhibit linear computational complexity in relation to the population, whereas GA and Firefly show quadratic complexity. This distinction becomes crucial when selecting an optimization algorithm for large-scale fog networks comprising dozens of nodes and thousands of services.

Third, based on the principle of architectural orthogonality — whereby GNN, MARL, and FL each solve a distinct subproblem without competing for the same optimization channel — the fully integrated MARL+GNN+FL architecture is theoretically projected to achieve the best combined profile across all five metrics: latency of 65–98 ms, energy efficiency improvement of up to 30% relative to greedy baselines, high scalability beyond $N = 50$ nodes, and guaranteed data privacy. This projection is supported by independently validated results for the MARL+GNN [10, 23] and FL [20, 21] sub-architectures in separate experimental environments. However, none of the 26 reviewed sources validates all three components in a single unified testbed; empirical co-validation therefore constitutes the primary direction for future experimental work.

Fourth, a formalised five-dimensional classification model $M: X \rightarrow Y$ has been constructed and verified against twenty-six primary sources. The model maps any IoT fog system configuration — defined by scale (S), workload variability (W), latency budget (L), privacy constraint (P), and topological complexity (T) — to one of six validated method classes via a priority-ordered decision function (Algorithm 1). The model is falsifiable, in that each mapping specifies the conditions under which it would be refuted by future experimental evidence; reproducible, in

that the same input vector always yields the same output; and extensible, in that new input dimensions can be appended without restructuring existing priorities.

Promising directions for further investigation include: developing adaptive versions of meta-heuristics for real-time fog applications; reducing the communication overhead of federated learning aggregation in heterogeneous fog networks; empirically co-validating the MARL+GNN+FL architecture within a unified simulation testbed (e.g., iFogSim or EdgeCloudSim); and standardising benchmarks to enable repeatable cross-method comparison under consistent experimental conditions.

ADDITIONAL INFORMATION

AUTHOR CONTRIBUTIONS

Conceptualization, L.V. and O.S.; methodology, L.V. and O.S.; formal analysis, O.S.; investigation, O.S.; writing – original draft preparation, O.S.; writing – review and editing, L.V. and O.S.; supervision, L.V.; project administration, L.V. All authors have read and agreed to the published version of the manuscript.

DECLARATION ON THE USE OF GENERATIVE ARTIFICIAL INTELLIGENCE TOOLS

In preparing this work, the authors used the following AI-assisted tools: DeepL Translate to check grammar and spelling; Claude (Anthropic) to assist with improving academic writing style; Perplexity AI to support literature search and verification of factual information across sources. After using these tools, the authors reviewed and edited the content and take full responsibility for the content of this publication.

REFERENCES

1. Srirama S. N. A Decade of Research in Fog Computing: Relevance, Challenges, and Future Directions. Software: Practice and Experience. 2024. Vol. 54, № 1. P. 3–23. DOI: <https://doi.org/10.1002/spc.3249>
2. Canali C., Lancellotti R. GASP: Genetic Algorithms for Service Placement in Fog Computing Systems. Algorithms. 2019. Vol. 12, № 10. P. 201. DOI: <https://doi.org/10.3390/a12100201>
3. Hussein M. K., Mousa M. H. Efficient Task Offloading for IoT-Based Applications in Fog Computing Using Ant Colony Optimization. IEEE Access. 2020. Vol. 8. P. 37191–37201. DOI: <https://doi.org/10.1109/ACCESS.2020.2975149>
4. Chen L., Guo S., Qiu X., Qi F., Li J., Guo Y. IoT Microservice Deployment in Edge-Cloud Hybrid Environment Using Reinforcement Learning. IEEE Internet of Things Journal. 2021. Vol. 8, № 16. P. 12610–12625. DOI: <https://doi.org/10.1109/JIOT.2020.3014970>
5. Azimi S., Pahl C., Shirvani M. H. Performance Management in Clustered Edge Architectures Using Particle Swarm Optimization. Cloud Computing and Services Science. CLOSER 2020 / eds. D. Ferguson, C. Pahl, M. Helfert. Cham : Springer, 2021. P. 187–208. (Communications in Computer and Information Science ; vol. 1399). DOI: https://doi.org/10.1007/978-3-030-72369-9_10
6. Subramoney D., Nyirenda C. N. Multi-Swarm PSO Algorithm for Static Workflow Scheduling in Cloud-Fog Environments. IEEE Access. 2022. Vol. 10. P. 117199–117214. DOI: <https://doi.org/10.1109/ACCESS.2022.3220239>
7. Pallewatta S., Kostakos V., Buyya R. QoS-Aware Placement of Microservices-Based IoT Applications in Fog Computing Environments. Future Generation Computer Systems. 2022. Vol. 131. P. 121–136. DOI: <https://doi.org/10.1016/j.future.2022.01.022>
8. Nguyen B. M., Binh H. T. T., Anh T. T., Son D. B. Evolutionary Algorithms to Optimize Task Scheduling Problem for the IoT Based Bag-of-Tasks Application in Cloud-Fog Computing Environment. Applied Sciences. 2019. Vol. 9, № 9. P. 1730. DOI: <https://doi.org/10.3390/app9091730>
9. Alghamdi T. A., Saad M., Enam R. N., Ul-Qayyum Z. Optimizing Multi-Objective Task Scheduling in Fog Computing with GA-PSO Algorithm for Big Data Application. Frontiers in Big Data. 2024. Vol. 7. P. 1358486. DOI: <https://doi.org/10.3389/fdata.2024.1358486>
10. Guerrero C., Lera I. Multi-Objective Application Placement in Fog Computing Using Graph Neural Network-Based Reinforcement Learning. The Journal of Supercomputing. 2024. Vol. 80, № 19. P. 27073–27094. DOI: <https://doi.org/10.1007/s11227-024-06439-5>
11. Garmendia-Orbegozo A., Nunez-Gonzalez J. D., Anton M. A. Task Offloading in Edge Computing Using GNNs and DQN. Computer Modeling in Engineering & Sciences. 2024. Vol. 139, № 3. P. 2649–2671. DOI: <https://doi.org/10.32604/cmcs.2024.045912>
12. Yousif A. An Adaptive Firefly Algorithm for Dependent Task Scheduling in IoT-Fog Computing. Computer Modeling in Engineering & Sciences. 2025. Vol. 142, № 3. P. 2869–2892. DOI: <https://doi.org/10.32604/cmcs.2025.059786>
13. Huang T., Lin W., Xiong C., Pan R., Huang J. An Ant Colony Optimization-Based Multiobjective Service Replicas Placement Strategy for Fog Computing. IEEE Transactions on Cybernetics. 2021. Vol. 51, № 11. P. 5595–5608. DOI: <https://doi.org/10.1109/TCYB.2020.2989309>

14. Mahapatra A., Majhi S. K., Mishra K., Pradhan R., Rao D. C., Panda S. K. An Energy-Aware Task Offloading and Load Balancing for Latency-Sensitive IoT Applications in the Fog-Cloud Continuum. IEEE Access. 2024. Vol. 12. P. 14334–14349. DOI: <https://doi.org/10.1109/ACCESS.2024.3357122>

15. Saif F. A., Latip R., Hanapi Z. M., Shafinah K., Kumar A. S., Bajaher A. S. Multi-Objectives Firefly Algorithm for Task Offloading in the Edge-Fog-Cloud Computing. IEEE Access. 2024. Vol. 12. P. 159561–159578. DOI: <https://doi.org/10.1109/ACCESS.2024.3488032>

16. Voloshchuk L., Sbitnev O. Hybrid Cloud-Based Intelligent Traffic Monitoring IoT System for a Residential Area. Advanced Information Technology. 2024. Vol. 1, № 3. P. 86–94. DOI: <https://doi.org/10.17721/AIT.2024.1.10>

17. Tu Y., Chen H., Yan L., Zhou X. Task Offloading Based on LSTM Prediction and Deep Reinforcement Learning for Efficient Edge Computing in IoT. Future Internet. 2022. Vol. 14, № 2. P. 30. DOI: <https://doi.org/10.3390/fi14020030>

18. Chen L., Zhang W., Ye H. Accurate Workload Prediction for Edge Data Centers: Savitzky-Golay Filter, CNN and BiLSTM with Attention Mechanism. Applied Intelligence. 2022. Vol. 52. P. 13027–13042. DOI: <https://doi.org/10.1007/s10489-021-03110-x>

19. Mirampalli S., Srirama S. N., Wankar R., Chillarige R. R. Hierarchical Fuzzy-Based Quality of Experience (QoE)-Aware Application Placement in Fog Nodes. Software: Practice and Experience. 2023. Vol. 53, № 2. P. 263–282. DOI: <https://doi.org/10.1002/spe.3146>

20. Kaleem S., Sohail A., Tariq M. U., Asim M. An Improved Big Data Analytics Architecture Using Federated Learning for IoT-Enabled Urban Intelligent

Transportation Systems. Sustainability. 2023. Vol. 15, № 21. P. 15333. DOI: <https://doi.org/10.3390/su152115333>

21. Hosseinalipour S., Azam S. S., Brinton C. G., Michelusi N., Aggarwal V., Love D. J., Dai H. Multi-Stage Hybrid Federated Learning over Large-Scale D2D-Enabled Fog Networks. IEEE/ACM Transactions on Networking. 2022. Vol. 30, № 4. P. 1569–1584. DOI: <https://doi.org/10.1109/TNET.2022.3142997>

22. Youn J., Han Y.-H. Intelligent Task Dispatching and Scheduling Using a Deep Q-Network in a Cluster Edge Computing System. Sensors. 2022. Vol. 22, № 11. P. 4098. DOI: <https://doi.org/10.3390/s22114098>

23. Jain V., Kumar B. QoS-Aware Task Offloading in Fog Environment Using Multi-agent Deep Reinforcement Learning. Journal of Network and Systems Management. 2023. Vol. 31, № 1. P. 7. DOI: <https://doi.org/10.1007/s10922-022-09696-y>

24. Mao W., Yin J., Liu Y., Cho B., Chen Y., Rao W., Xiao Y. Multi-Agent Reinforcement Learning-Based Capacity Planning for On-Demand Vehicular Fog Computing. IEEE Transactions on Intelligent Vehicles. 2024. DOI: <https://doi.org/10.1109/TIV.2024.3422375>

25. Müller P., Leimo M., Bittencourt L., Prodan R. Deep Reinforcement Learning Multi-Agent System for Resource Allocation in Industrial Internet of Things. Sensors. 2022. Vol. 22, № 13. P. 4788. DOI: <https://doi.org/10.3390/s22134788>

26. Xu A., Zhang X., Cui H., Chen B., Shi Y., Liu X. TransEdge: Task Offloading with GNN and DRL. IEEE Internet of Things Journal. 2024. Vol. 11, № 23. P. 38151–38165. DOI: <https://doi.org/10.1109/JIOT.2024.3397349>

Олександр СБІТНЄВ, Людмила ВОЛОЩУК
 Одеський національний університет ім. І. І. Мечникова

ПОРІВНЯЛЬНИЙ АНАЛІЗ МЕТОДІВ ОПТИМІЗАЦІЇ АРХІТЕКТУРИ FOG-СИСТЕМ: ВІД МЕТАЕВРИСТИК ДО ГІБРИДНИХ РІШЕНЬ

Стрімке поширення пристроїв Інтернету речей посилює вимоги до розподілених обчислювальних інфраструктур, роблячи туманні обчислення — парадигму, що розміщує обчислювальні ресурси на межі мережі — критично важливим засобом забезпечення низьколатентних застосунків реального часу. Оптиміальне розміщення сервісів на туманних вузлах є доведеною NP-складною комбінаторною задачею. Попри десятиліття активних досліджень, наявні роботи розглядають окремі родини алгоритмів ізольовано, залишаючи розробників систем без обґрунтованої бази для вибору методу оптимізації, придатного для конкретного контексту розгортання. Мета цієї статті — провести систематичний порівняльний аналіз понад десяти підходів до оптимізації в єдиній аналітичній рамці та побудувати формалізовану п'ятивимірну класифікаційну модель $M: X \rightarrow Y$ ($X = S \times W \times L \times P \times T$), що відображає характеристики IoT-системи на найбільш придатний клас методів. Дослідження узагальнює результати двадцяти шести первинних джерел за трьома класами методів — метаевристичні алгоритми (GA, PSO, ACO, Firefly), методи машинного навчання (навчання з учителем, LSTM, CNN-BiLSTM, GNN, федеративне навчання) та навчання з підкріпленням (одноагентний DQN і багатоагентний MARL) — за п'ятьма метриками: затримка обробки, енергоефективність, масштабованість, адаптивність до динамічних навантажень та складність реалізації. Аналіз дає три основні результати. По-перше, PSO та ACO демонструють лінійну складність $O(n)$ на ітерацію, тоді як GA та Firefly — квадратичну $O(n^2)$, через що складність стає вирішальним критерієм вибору для туманних мереж понад 50 вузлів: на цьому порозі $T_{GA} \approx 100 \cdot T_{PSO}$. По-друге, федеративне навчання досягає точності класифікації понад 93% у транспортних IoT-системах без централізації первинних даних і є єдиним класом методів, що задовольняє законодавчі вимоги щодо суверенітету даних. По-третє, повністю інтегрований підхід MARL+GNN+FL теоретично прогнозується досягти затримки 65–98 мс і покращення енергоефективності на 22–30% відносно жадібних базових методів, виходячи з принципу архітектурної ортогональності та незалежності підтверджених результатів для підархітектур MARL+GNN і FL; проте жодне з 26 розглянутих джерел не валідую цю комбіновану

архітектуру в єдиному стенді, і емпірична спільна валідація є основним напрямом майбутніх досліджень. Формалізована модель $M: X \rightarrow Y$ включає явні числові порогові значення для всіх п'яти вхідних вимірів, пріоритетно впорядковану функцію прийняття рішень (Algorithm 1) і шість верифікованих вихідних класів; модель є фальсифікованою, відтворюваною та розширюваною, а її застосування демонструється на сценарії розгортання туманної мережі смарт-міста з $N = 75$ вузлів.

Ключові слова: туманні обчислення, метаввристична оптимізація, навчання з підкріпленням, гібридні методи штучного інтелекту, оптимізація ресурсів, розміщення сервісів.