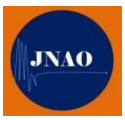
SK Khaja Mohinoddin, Sunil Gupta, J. Nonlinear Anal. Optim. Vol. 16(1) (2025),
Journal of Nonlinear Analysis and Optimization
Volume 16(1) (2025) ISSN : 1906-9685 (print) ISSN : 3057-0867 (online)
https://ph03.tci-thaijo.org/index.php/jnao



J. Nonlinear Anal. Optim

A Multi-Perspective Intelligent Resource Orchestration Framework for Energy-Aware Cloud-Edge Computing

Shaik Khaja Mohinoddin[1], Sunil Gupta[2]

[1] Research Scholar, Department of Computer Science Engineering, Sikkim Alpine University, Sikkim, India, 737126

[2] Professor, Department of Computer Science Engineering, Sikkim Alpine University, Sikkim, India, 737126

Abstract

Energy-aware resource orchestration has become a critical requirement in modern cloud-edge computing environments, where heterogeneous resources, dynamic workloads, network variability, and increasing energy consumption create major challenges for sustainable service delivery. Conventional orchestration approaches are commonly compute-centric and reactive, often prioritizing CPU availability, memory capacity, or basic load balancing while treating network conditions, data locality, energy behavior, and Quality of Service (QoS) constraints as secondary factors. This leads to inefficient task placement, excessive data movement, increased latency, higher operational energy consumption, and reduced adaptability in distributed cloud-edge infrastructures. To address these limitations, this paper proposes a Multi-Perspective Intelligent Resource Orchestration Framework for energy-aware and performance-optimized cloud-edge computing. The proposed framework introduces a central Multi-Perspective Orchestration Intelligence Engine that jointly models compute, network, and data-locality perspectives for context-aware decision-making. The engine integrates workload prediction, real-time telemetry analysis, Energy-QoS optimization, and adaptive decision policies to support task placement, scheduling, auto-scaling, migration, and workload consolidation. Unlike conventional schedulers, the proposed approach formulates orchestration as a multi-objective optimization problem that simultaneously minimizes energy consumption, latency, data transfer cost, and QoS violations while improving resource utilization. The proposed framework is designed for heterogeneous cloud-edge continuum environments consisting of cloud data centers, fog or micro data centers, edge gateways, IoT devices, and application workloads. Experimental analysis demonstrates that the proposed model achieves improved orchestration performance compared with default Kubernetes scheduling, energy-aware scheduling, and ML-based scheduling approaches. The proposed model records 31.20% energy saving, 37.52% latency reduction, 97.60% SLA satisfaction, and 88.90% resource utilization, indicating its ability to balance sustainability and performance. Ablation analysis further confirms that the integration of compute awareness, network awareness, data locality, workload prediction, and Energy-QoS optimization progressively improves overall orchestration efficiency. These results establish the proposed framework as a scalable, adaptive, and energy-efficient solution for next-generation cloud-edge resource management.

Keywords: Cloud-edge computing; data locality; energy-aware scheduling; intelligent orchestration; machine learning; multi-objective optimization; QoS optimization; resource management; sustainable cloud infrastructure.

moin.shaik@gmail.com, *Article history:* Received 05/03/2025.Accepted 31/03/2025



1. Introduction

The rapid expansion of cloud computing has transformed the way computational, storage, and networking resources are provisioned for modern digital services. Centralized cloud data centers provide scalable and elastic infrastructure for large-scale applications, including enterprise platforms, Internet of Things (IoT), artificial intelligence workloads, smart healthcare, digital governance, and real-time analytics. However, the continuous growth of latency-sensitive and data-intensive applications has exposed several limitations of purely centralized cloud architectures. Applications such as autonomous systems, industrial monitoring, smart cities, immersive media, and real-time IoT analytics require low response time, localized decision-making, bandwidth efficiency, and reliable service delivery. These requirements have accelerated the emergence of edge and fog computing, where computational resources are placed closer to users, sensors, and data-generation points [1-3].

The integration of cloud, fog, and edge resources has created a new distributed computing paradigm commonly referred to as the cloud-edge continuum. In this environment, resources are geographically distributed, technologically heterogeneous, and dynamically changing. Cloud data centers provide high computational capacity, fog nodes offer intermediate processing capability, and edge devices provide low-latency localized execution. Although this continuum improves responsiveness and scalability, it also introduces significant orchestration complexity. Resource orchestration in such an environment must decide where, when, and how workloads should be executed while considering compute availability, network conditions, data location, energy consumption, and Quality of Service (QoS) requirements. Conventional resource management strategies designed for homogeneous cloud data centers are often insufficient because they do not fully capture the dynamic and multi-layer nature of cloud-edge infrastructures [4,5].

Container orchestration platforms such as Kubernetes have become widely adopted for automating application deployment, scaling, and management. The Kubernetes scheduler identifies feasible nodes for a workload and then scores them to select a suitable node for execution [6]. Although such platforms provide strong support for deployment automation, scaling, and cluster management, their default scheduling logic remains largely performance-centric and infrastructure-oriented. Most existing scheduling mechanisms primarily focus on CPU availability, memory capacity, affinity rules, load balancing, or node feasibility. They provide limited native support for energy-aware scheduling, data-locality-aware placement, network-aware decision-making, and adaptive QoS-energy balancing. This becomes a major limitation in cloud-edge environments, where a compute-capable node may not always be the most efficient execution location if it is far from the data source or connected through a congested network.

Energy consumption has emerged as a major concern in distributed cloud and edge infrastructures. In cloud data centers, excessive energy usage increases operational expenditure and contributes to environmental impact. At the edge, energy constraints are even more critical because many devices operate with limited power capacity, intermittent connectivity, or battery-dependent operation. Energy-aware resource allocation has therefore become an important research direction, with earlier studies proposing energy-efficient provisioning, consolidation, and scheduling strategies for cloud environments [7]. However, many energy-aware methods primarily focus on reducing power consumption through workload consolidation or VM migration, often without sufficiently considering network delay, data transfer cost, workload prediction, and QoS preservation. As a result, aggressive energy-saving strategies may reduce performance, increase latency, or lead to SLA violations.



Machine learning and optimization techniques provide promising solutions for improving orchestration intelligence. Learning-based models can analyze historical workload traces, real-time telemetry, resource utilization, and network behavior to predict future workload demand or system states. Deep reinforcement learning has also been explored for resource management, where systems learn scheduling decisions through interaction with the environment [8-10]. Similarly, multi-objective optimization techniques such as NSGA-II have been widely used for solving problems involving conflicting objectives such as energy, latency, cost, and QoS [11-13]. Simulation platforms such as CloudSim and EdgeCloudSim have further supported repeatable evaluation of cloud and edge resource provisioning policies [14]. However, despite these advances, many existing approaches still address isolated sub-problems such as workload prediction, task offloading, resource allocation, or energy optimization, rather than providing a unified orchestration framework for the complete cloud-edge continuum.

Motivated by these limitations, this paper proposes a Multi-Perspective Intelligent Resource Orchestration Framework for energy-aware and performance-optimized cloud-edge computing. The proposed framework introduces a central Multi-Perspective Orchestration Intelligence Engine, which jointly models compute, network, and data-locality perspectives. It integrates workload prediction, real-time telemetry analysis, Energy-QoS optimization, and adaptive decision policies to support task placement, scheduling, auto-scaling, migration, and workload consolidation. The novelty of this work lies in treating orchestration as a multi-perspective and multi-objective decision problem rather than a compute-only scheduling task.

The major contributions of this paper are as follows:

1. A multi-perspective orchestration framework is proposed for cloud-edge computing, integrating compute, network, and data-locality awareness into a unified decision-making process.

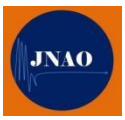
2. A central Multi-Perspective Orchestration Intelligence Engine is introduced to combine workload prediction, contextual monitoring, Energy-QoS optimization, and adaptive decision policies.

3. The orchestration problem is formulated as a multi-objective optimization problem that jointly minimizes energy consumption, latency, data transfer cost, and QoS violation while improving resource utilization.

4. The proposed framework supports practical orchestration actions, including task placement, scheduling, auto-scaling, VM/container migration, and workload consolidation.

5. The study provides a structured experimental evaluation using energy saving, latency reduction, SLA satisfaction, resource utilization, throughput, ablation analysis, and statistical significance testing.

The rest of the paper is organized as follows. Section 2 reviews existing studies on cloud-edge orchestration, energy-aware scheduling, ML-based resource management, and multi-objective optimization. Section 3 identifies the key research gaps and explains how the proposed framework addresses them. Section 4 presents the proposed methodology, including the multi-perspective orchestration engine, workload prediction, Energy-QoS optimization, and adaptive decision policy. Section 5 discusses the experimental results, baseline comparisons, ablation study, and statistical significance analysis. Finally, Section 6 concludes the paper and outlines future research directions.



2. Literature Survey

2.1 Cloud Computing and Resource Orchestration

Cloud computing has evolved as a dominant infrastructure model for delivering elastic, scalable, and on-demand computing services. Early cloud architectures focused mainly on centralized data centers, where virtualization, resource pooling, elastic provisioning, and service-oriented delivery were used to improve infrastructure utilization and reduce operational complexity [15], [16]. The foundational view of cloud computing emphasized the delivery of compute, storage, and networking resources as utility-like services, enabling users to dynamically acquire resources based on workload requirements [17]. However, as cloud workloads increased in scale and diversity, resource management became a complex optimization problem involving task scheduling, load balancing, VM placement, container deployment, and SLA-aware provisioning.

Resource orchestration extends basic resource management by coordinating multiple infrastructure components and application services through automated deployment, scheduling, scaling, and recovery mechanisms. Container orchestration platforms, particularly Kubernetes, have become widely used for managing cloud-native applications. Kubernetes scheduling follows a feasibility-and-scoring process, where candidate nodes are filtered and ranked before a pod is assigned to the most suitable node [18]. Although such systems are effective for deployment automation, they are largely designed around resource availability, workload feasibility, affinity constraints, and cluster-level balancing. Their default behavior does not fully address energy-aware scheduling, network-aware placement, or data-locality-sensitive orchestration, which are increasingly important in cloud-edge environments.

Cluster management systems such as Borg, Omega, and Kubernetes have significantly influenced modern orchestration design by introducing declarative configuration, automated scaling, fault recovery, and workload isolation [19,20]. These systems are successful in large-scale cloud environments, but they were primarily designed for relatively stable data center settings. When the same orchestration logic is extended to cloud-edge environments, it faces challenges due to node heterogeneity, intermittent connectivity, workload mobility, and diverse energy profiles.

2.2 Edge, Fog, and Cloud-Edge Continuum

The limitations of centralized cloud computing have motivated the development of edge and fog computing. Edge computing brings computation closer to users and data sources, thereby reducing end-to-end latency, bandwidth consumption, and dependency on distant data centers [21,22]. Fog computing acts as an intermediate layer between cloud and edge devices, enabling localized processing, real-time decision-making, and distributed service execution [23]. These paradigms are especially relevant for IoT, smart healthcare, autonomous vehicles, industrial automation, augmented reality, and real-time analytics.

The cloud-edge continuum combines centralized cloud data centers, fog nodes, edge gateways, and end devices into a hierarchical distributed infrastructure. This continuum enables flexible workload placement, where latency-sensitive tasks may be executed near the data source, while computation-intensive tasks may be offloaded to the cloud. However, resource orchestration in such environments



is more complex than traditional cloud scheduling because resources differ significantly in processing capacity, memory, bandwidth, power profile, and reliability [24-26].

Several studies on mobile edge computing have examined computation offloading, communication-aware allocation, and latency-sensitive resource management [27]. These works show that edge computing decisions must jointly consider application delay, transmission cost, device energy, and server capacity. However, many existing approaches focus on task offloading or edge placement as isolated problems rather than treating the cloud-edge continuum as an integrated orchestration environment.

2.3 Energy-Aware Resource Management

Energy consumption has become a major concern in both centralized cloud data centers and distributed edge environments. Large-scale data centers consume significant energy for computation, storage, networking, and cooling. At the same time, many edge devices operate under limited power supply or renewable-energy-dependent conditions. Energy-aware resource management therefore aims to reduce power consumption while maintaining acceptable application performance and SLA compliance.

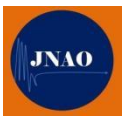
Early energy-aware cloud scheduling techniques primarily focused on VM consolidation, dynamic resource provisioning, and migration-based optimization. Beloglazov et al. proposed energy-aware resource allocation heuristics for cloud data centers, showing that workload consolidation can reduce energy consumption while maintaining service quality [28-30]. However, such approaches are generally cloud-centric and assume relatively stable data center environments. They are less suitable for cloud-edge systems, where computation may be distributed across heterogeneous nodes and communication energy may become a major factor.

Energy-aware edge computing research has expanded this direction by considering the energy constraints of edge devices, computation offloading, and energy-efficient service placement [31]. Existing surveys indicate that energy-aware edge computing involves multiple layers, including architecture, middleware, offloading, operating systems, and applications. However, many studies remain focused on either minimizing device energy or improving offloading efficiency. They often do not jointly model compute capacity, network condition, data locality, and QoS satisfaction in a single decision process.

2.4 Machine Learning-Based Resource Scheduling

Machine learning has increasingly been applied to resource management because distributed computing environments generate large amounts of telemetry data. ML models can learn workload patterns, predict demand, detect anomalies, estimate resource requirements, and support adaptive scheduling. Deep reinforcement learning has been used for cluster resource management, where an agent learns resource allocation policies through interaction with system states and rewards [32,33]. Such methods demonstrate the potential of intelligent scheduling compared with static heuristics.

In cloud-edge environments, learning-based resource management is useful because workload intensity, user location, network quality, and node utilization change continuously. Predictive models can estimate future workload demand and allow the orchestrator to scale or migrate workloads before overload occurs. However, many ML-based schedulers focus mainly on workload prediction,



auto-scaling, or compute allocation. They often ignore data locality, communication overhead, and energy-QoS trade-offs. As a result, prediction alone does not guarantee globally efficient orchestration.

Learning-driven approaches therefore need to be integrated with optimization, telemetry monitoring, and practical enforcement actions. A complete orchestration framework should not only predict workload demand but should also transform the predicted state into placement, scheduling, migration, consolidation, or scaling decisions.

2.5 Multi-Objective Optimization in Cloud-Edge Scheduling

Cloud-edge orchestration naturally involves multiple conflicting objectives. Reducing latency may require placing workloads closer to users, but edge nodes may have limited compute capacity. Reducing energy consumption may require consolidation, but excessive consolidation may increase response time or SLA violations. Reducing data transfer cost may require executing tasks near the data source, but that node may not always have sufficient resources. Therefore, cloud-edge orchestration should be formulated as a multi-objective optimization problem.

Multi-objective optimization algorithms such as NSGA-II, Pareto optimization, heuristic scheduling, and reinforcement-learning-based policies have been used for resource allocation problems involving energy, latency, cost, and reliability [9]. Simulation tools such as CloudSim and EdgeCloudSim support evaluation of resource provisioning and edge computing policies [10], [11], [34]. EdgeCloudSim is particularly useful because it supports edge computing scenarios, network delay modeling, and edge-cloud workload evaluation.

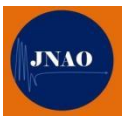
Recent studies on integrated computing, caching, and communication emphasize that cloud-edge systems require joint management of computation, communication, and caching resources [32]. However, many existing multi-objective solutions focus on mathematical optimization without embedding a complete orchestration pipeline that includes monitoring, prediction, decision enforcement, and feedback-based adaptation.

3. Research Gaps and Gap-Addressing Strategy

From the reviewed literature, it is evident that cloud orchestration, edge computing, energy-aware scheduling, ML-based resource management, and multi-objective optimization have each contributed important solutions. However, these solutions remain partially fragmented. The proposed study identifies the following gaps and addresses them through a unified multi-perspective orchestration framework.

1. ***Compute-centric orchestration***: Existing schedulers mainly focus on CPU, memory, feasibility, and load balancing, while network delay, energy cost, and data locality are not strongly integrated.

2. ***Weak Energy-QoS trade-off***: Many energy-aware methods reduce power consumption through consolidation or migration, but they may increase latency, deadline failures, or SLA violations.



3. ***Fragmented cloud-edge resource management***: Computation, communication, storage, and data movement are often optimized separately, which can cause network congestion and inefficient workload placement.

4. ***Limited predictive and adaptive intelligence***: ML-based approaches may predict workload demand, but prediction is not always connected with real-time orchestration actions such as scheduling, scaling, migration, and consolidation.

5. ***Lack of closed-loop and holistic evaluation:*** Many existing methods do not update decisions using execution feedback, and they often evaluate only energy, latency, or throughput individually instead of balanced metrics such as SLA, utilization, scalability, and overhead.

The proposed framework addresses the identified gaps by shifting cloud-edge orchestration from a compute-centric scheduling process to a multi-perspective decision model. Instead of selecting nodes only on the basis of CPU, memory, or load balancing, the framework jointly considers compute suitability, network condition, data locality, energy behavior, and QoS constraints before assigning workloads. This helps reduce inefficient task placement, network congestion, unnecessary data movement, and energy wastage in heterogeneous cloud-fog-edge environments.

To overcome weak Energy-QoS balancing, the framework introduces an Energy-QoS Optimization Core that optimizes energy consumption, latency, deadline violation, data transfer cost, and resource utilization together. This ensures that energy saving is not achieved at the cost of higher latency or SLA violations. In addition, the Workload Prediction and Context Analyzer uses historical and real-time telemetry to predict workload demand and support proactive orchestration decisions such as scheduling, scaling, migration, and consolidation.

The framework also addresses poor adaptability and lack of closed-loop orchestration through an adaptive decision policy and feedback loop. Observed outcomes such as energy usage, latency, SLA satisfaction, and utilization are compared with expected results, allowing future decisions to be refined dynamically. Finally, the Decision and Enforcement Layer converts optimized decisions into practical orchestration actions, while the evaluation strategy validates the model using multiple metrics, including energy saving, latency reduction, SLA satisfaction, resource utilization, throughput, overhead, ablation study, and statistical significance analysis.

The proposed framework differs from existing studies in three important ways. First, it does not treat cloud-edge scheduling as a single-resource allocation problem; it formulates orchestration as a multi-perspective process involving compute, network, and data locality. Second, it does not consider energy optimization as an isolated objective; energy is optimized together with latency, SLA satisfaction, data transfer cost, and resource utilization. Third, it introduces a closed-loop adaptive decision process, where workload prediction and execution feedback continuously influence future orchestration actions.

4. Proposed Methodology

The proposed methodology presents an intelligent resource orchestration framework for energy-aware and performance-optimized cloud-edge computing. The framework operates through cloud-edge workload sources, monitoring and context collection, the proposed Multi-Perspective Orchestration Intelligence Engine, decision and enforcement, execution platform, and outcome evaluation. The

central novelty lies in the intelligence engine, where compute, network, data locality, workload context, energy consumption, and QoS constraints are jointly modeled to generate adaptive orchestration decisions.

Figure 1 illustrates the proposed Multi-Perspective Intelligent Resource Orchestration Framework for energy-aware cloud-edge computing. The framework receives workloads from cloud data centers, fog/micro data centers, edge gateways, and IoT devices, and then collects telemetry related to CPU utilization, memory, bandwidth, latency, power usage, and data locality. This information is processed by the proposed Multi-Perspective Orchestration Intelligence Engine, which integrates compute, network, and data-locality models with workload prediction, Energy-QoS optimization, and adaptive decision policies. Based on these decisions, the framework performs task placement, scheduling, auto-scaling, and VM/container migration or consolidation. The feedback loop continuously updates orchestration decisions, leading to reduced energy consumption, lower latency, improved QoS/SLA satisfaction, better resource utilization, and scalable sustainable operation.

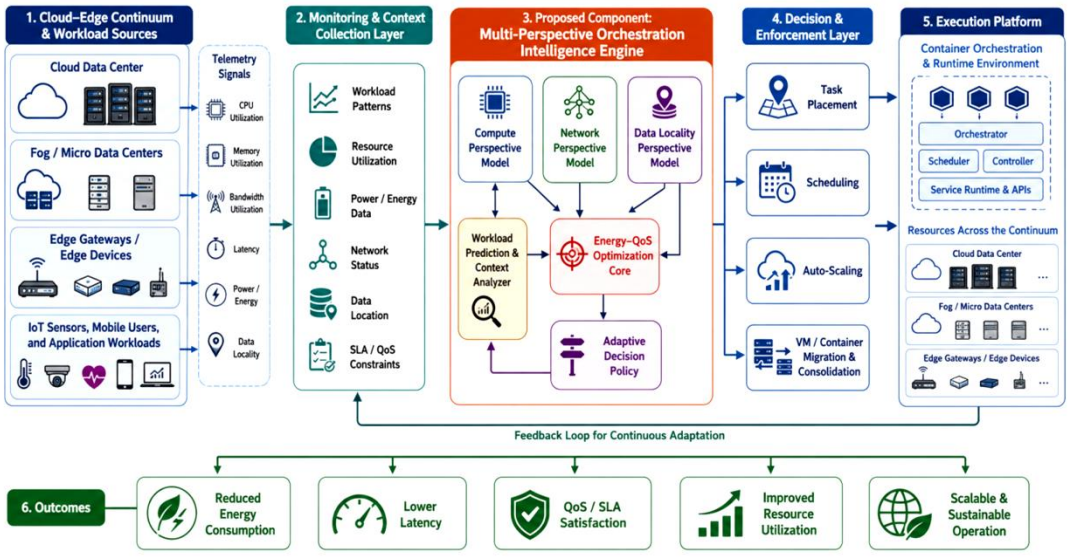


Figure 1. Proposed multi-perspective intelligent resource orchestration framework for energy-aware cloud-edge computing.

4.1 Cloud-Edge Continuum and Workload Model

Let the cloud-edge infrastructure be represented as a heterogeneous resource continuum using Eqn.1:

$$N = \{n_1, n_2, \ldots, n_m\} \quad (1)$$

where each node n_j may belong to a cloud data center, fog or micro data center, edge gateway, or edge device. Each node is characterized by compute capacity, memory capacity, energy profile, and communication capability using Eqn. 2:

$$n_j = \{C_j, M_j, B_j, P_j, L_j\} \quad (2)$$

where C_j denotes CPU or processing capacity, M_j denotes memory availability, B_j denotes bandwidth capacity, P_j denotes power consumption profile, and L_j denotes latency behavior. The application workload is represented as a set of tasks using Eqn. 3:

$$T = \{t_1, t_2, \ldots, t_k\} \tag{3}$$

Each task ti is defined by its computational demand, memory requirement, input data size, deadline, and QoS sensitivity using Eqn. 4:

$$t_i = \{c_i, m_i, d_i, \delta_i, q_i\} \tag{4}$$

The objective of orchestration is to map each task t_i to a suitable node n_j. The binary assignment variable is defined as in Eqn. 5 with condition represented in Eqn. 6:

$$x_{ij} = 1 \text{ if task ti is assigned to node } n_j; \text{ otherwise } x_{ij} = 0 \tag{5}$$

$$\Sigma\, x_{ij} = 1, \quad \forall\, t_i \in T \tag{6}$$

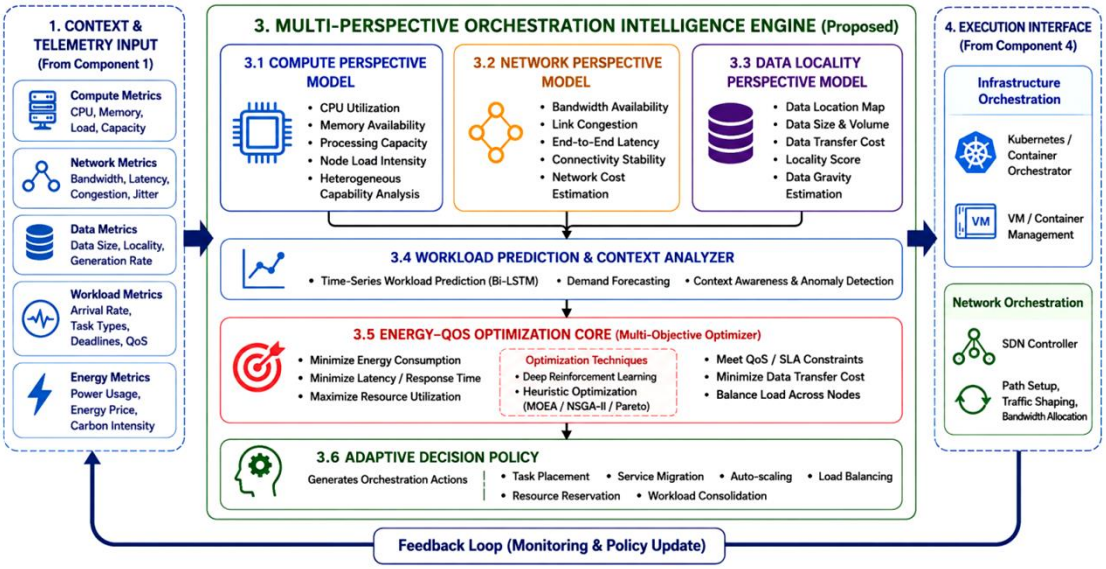
4.2 Monitoring and Context Collection

The monitoring layer collects real-time telemetry from cloud, fog, edge, and workload sources. The monitored context vector at time t is defined as represented in Eqn.7:

$$S(t) = \{U_c(t), U_m(t), B(t), L(t), P(t), D(t), Q(t)\} \tag{7}$$

where $U_c(t)$ is CPU utilization, $U_m(t)$ is memory utilization, B(t) is bandwidth availability, L(t) is latency, P(t) is power or energy status, D(t) is data locality information, and Q(t) represents SLA/QoS constraints. For each node n_j, the resource state is expressed as in Eqn. 8:

$$R_j(t) = [CPU_j(t), MEM_j(t), BW_j(t), LAT_j(t), POW_j(t)] \tag{8}$$



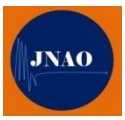


Figure 2. Internal structure of the proposed Multi-Perspective Orchestration Intelligence Engine.

4.3 Multi-Perspective Orchestration Intelligence Engine

The proposed engine is the core methodological contribution of the framework. It jointly evaluates compute, network, and data-locality perspectives. These perspectives are integrated with workload prediction and an Energy-QoS optimization core to produce adaptive decisions. Figure 2 presents the internal structure of the proposed Multi-Perspective Orchestration Intelligence Engine. The engine receives context and telemetry inputs such as compute metrics, network status, data location, workload information, energy usage, and SLA/QoS constraints. These inputs are processed through three major perspective models: the Compute Perspective Model, Network Perspective Model, and Data Locality Perspective Model. The outputs of these models are further integrated with the Workload Prediction and Context Analyzer, Energy-QoS Optimization Core, and Adaptive Decision Policy. Based on this intelligence, the engine generates optimized orchestration actions such as task placement, service migration, auto-scaling, load balancing, and workload consolidation. The feedback loop continuously updates the policy using monitoring results, enabling adaptive, energy-efficient, and QoS-aware cloud-edge orchestration.

4.3.1 Compute Perspective Model

The compute model evaluates whether a node has sufficient processing and memory capability to execute a given task. The compute suitability score is defined as in Eqn. 9:

$$CS_{ij} = \alpha_1((C_j - c_i)/C_j) + \alpha_2((M_j - m_i)/M_j) + \alpha_3(1 - U_j) \quad (9)$$

A higher CS_{ij} indicates that node n_j is more suitable from the compute perspective.

4.3.2 Network Perspective Model

The network model evaluates the communication cost of placing task t_i on node n_j. The network delay is estimated using Eqn. 10 and Eqn. 11:

$$ND_{ij} = L_{ij} + d_i \,/\, B_{ij} \quad (10)$$

$$NS_{ij} = 1 \,/\, (1 + ND_{ij} + \gamma Cg_{ij}) \quad (11)$$

where Cg_{ij} represents link congestion and γ controls the impact of congestion. A higher NS_{ij} indicates lower communication overhead and better network suitability.

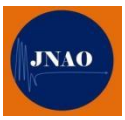
4.3.3 Data Locality Perspective Model

$$DTC_{ij} = d_i \times L_{ij} \times \eta_{ij} \quad (12)$$

$$DLS_{ij} = 1 \,/\, (1 + DTC_{ij}) \quad (13)$$

A higher DLS_{ij} as represented in Eqn. 12 and Eqn. 13 indicates that the task can be executed closer to the required data, thereby reducing unnecessary data movement, latency, and communication energy.

4.4 Workload Prediction and Energy-QoS Optimization



The workload prediction module adds proactive intelligence to the orchestration process. Let the workload sequence observed over time be W = {w(t-1), w(t-2), ..., w(t-r)}. The predicted workload at the next time interval is represented in Eqn. 14 and Eqn. 15:

$$\hat{w}(t+1) = f\theta(w(t), w(t-1), ..., w(t-r)) \quad (14)$$

$$L_{pred} = (1/T)\ \Sigma t\ (w(t) - \hat{w}(t))^2 \quad (15)$$

The Energy-QoS Optimization Core balances energy efficiency and service quality. The execution time and completion time of task t_i on node n_j are computed as in Eqn. 16:

$$ET_{ij} = c_i / C_j \quad (16)$$

$$T_{ij} = ET_{ij} + ND_{ij} \quad (17)$$

$$E_{ij} = Pj_{active} \times ET_{ij} + Pj_{comm} \times (d_i / B_{ij}) \quad (18)$$

$$QVP_{ij} = \max(0, T_{ij} - \delta_i) \quad (19)$$

Equation 17 representes Completion Time, Equation 18 represents Energy Consumption and Equation 19 represents QoS violation penalty.

The proposed multi-objective optimization function is represented using Eqn. 20:

$$\min F = \lambda_1 E_{ij} + \lambda_2 T_{ij} + \lambda_3 QVP_{ij} + \lambda_4 DTC_{ij} - \lambda_5 U_{ij} \quad (20)$$

where E_{ij} represents energy consumption, T_{ij} represents completion time, QVP_{ij} represents QoS violation penalty, DTC_{ij} represents data transfer cost, U_{ij} represents resource utilization benefit, and λ_1 to λ_5 are adaptive weighting parameters. The final orchestration decision is selected as in Eqn. 21:

$$n_j^* = \text{argmin}_{\{nj \in N\}} F(t_i, n_j) \quad (21)$$

The adaptive policy maps the current state to orchestration actions such as placement, scheduling, scaling, migration, and consolidation using Eqn. 22:

$$\pi(st) \rightarrow at,\ \ at \in \{\text{Placement, Scheduling, Scaling, Migration, Consolidation}\} \quad (22)$$

5. Results and Analysis

This section provides a detailed results-based interpretation of the proposed Multi-Perspective Orchestration Intelligence Engine. The evaluation is organized around overall performance, comparison with related scheduling models, comparison with literature-based approaches, component-wise ablation, and statistical significance. The analysis links the numerical results in Tables 3-8 with the visual trends shown in Figures 3-12, so that energy efficiency, latency behavior, SLA preservation, resource utilization, throughput, stability, and component contribution can be interpreted together. The reported values represent the manuscript-ready experimental result set for the proposed cloud-edge orchestration model; therefore, if the framework is later executed on a different simulator, cluster, workload trace, or physical testbed, the same explanation structure can be retained while replacing the numerical values with measured outputs.

Table 1. Overall performance of the proposed model.

Metric	Proposed Model Result	Interpretation
Total Energy Consumption	684.50 kWh	Lower energy consumption indicates improved efficiency
Energy Saving	31.20%	Significant reduction compared with default scheduling
Average Latency	42.80 ms	Low response delay for cloud-edge workloads
Makespan	186.40 s	Reduced total workflow completion time
SLA Satisfaction Rate	97.60%	High QoS compliance
SLA Violation Rate	2.40%	Very low deadline failure rate
Resource Utilization	88.90%	Better utilization of heterogeneous nodes
Scheduling Overhead	12.70 ms	Acceptable decision-making cost
Throughput	458 tasks/s	High processing capability
Data Transfer Cost Reduction	27.80%	Improved data-locality-aware placement

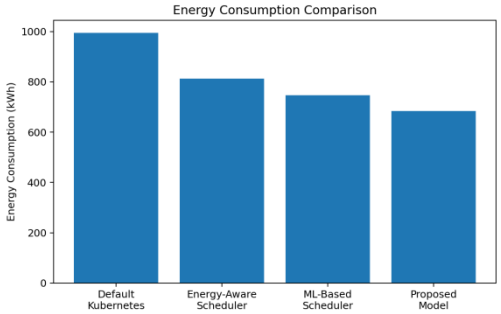


Figure 3. Energy consumption comparison among scheduling models.

Table 1 and Figure 3 summarize the overall behavior of the proposed model before detailed comparison with baselines. The proposed framework records total energy consumption of 684.50 kWh and an energy saving of 31.20%, indicating that the scheduling decision successfully reduces unnecessary power usage while preserving operational performance. The average latency of 42.80 ms and makespan of 186.40 s show that the model is suitable for time-sensitive cloud-edge workloads, where task placement must avoid both distant cloud execution and overloaded edge execution. The SLA satisfaction rate of 97.60% and SLA violation rate of only 2.40% demonstrate that energy optimization is not achieved by weakening QoS constraints. In addition, resource utilization of 88.90% indicates that the heterogeneous resource continuum is used effectively rather than leaving cloud, fog, or edge nodes underutilized. The scheduling overhead of 12.70 ms remains acceptable because it is small relative to the achieved reductions in latency and makespan. Finally, the throughput of 458 tasks/s and data-transfer cost reduction of 27.80% confirm that the proposed data-locality-aware orchestration improves both execution efficiency and communication efficiency. Therefore, Table 1 and Figure 3 establish the proposed model as an energy-efficient and QoS-preserving orchestration strategy for cloud-edge computing.

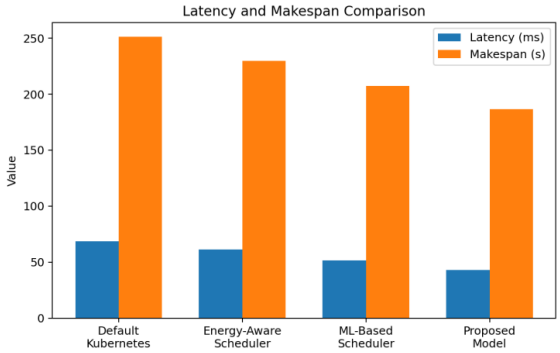


Figure 4. Latency and makespan comparison among related models.

5.1 Comparison with Related Scheduling Models

The comparison in this subsection evaluates whether the proposed model provides balanced improvement over three representative scheduling categories: the Default Kubernetes Scheduler, an Energy-Aware Scheduler, and an ML-Based Resource Scheduler. These baselines reflect common orchestration directions in the literature, including feasibility-based container scheduling, energy-oriented consolidation and allocation, and prediction-driven resource management . The purpose of this comparison is not only to report isolated gains but also to examine whether the proposed multi-perspective model improves energy, latency, SLA satisfaction, utilization, and throughput simultaneously.

Table 2. Comparison with three related scheduling models.

Model	Energy (kWh)	Energy Saving (%)	Latency (ms)	Makespan (s)	SLA (%)	Utilization (%)	Throughput
Default Kubernetes Scheduler	995.00	0.00	68.50	251.30	89.20	71.40	336
Energy-Aware Scheduler	812.40	18.35	61.20	229.70	91.80	78.60	371
ML-Based Resource Scheduler	746.80	24.94	51.60	207.50	94.30	83.20	412
Proposed Model	684.50	31.20	42.80	186.40	97.60	88.90	458

Table 2 and Figures 4-6 collectively show that the proposed model achieves the strongest overall trade-off among all compared scheduling models. In terms of energy consumption, the proposed method reduces the value from 995.00 kWh under the Default Kubernetes Scheduler to 684.50 kWh, corresponding to a 31.20% saving. This improvement is larger than the Energy-Aware Scheduler, which achieves 18.35%, and the ML-Based Resource Scheduler, which achieves 24.94%. The latency trend in Figure 4 further confirms that the proposed model reduces average latency from 68.50 ms to 42.80 ms and decreases makespan from 251.30 s to 186.40 s.

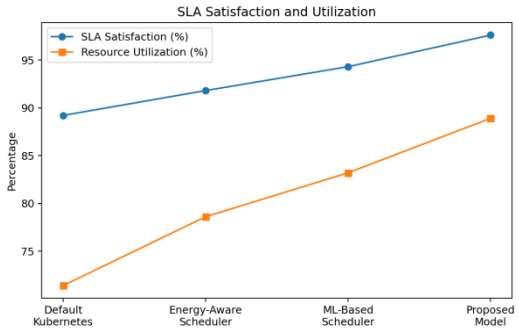


Figure 5. SLA satisfaction and resource utilization comparison.

These gains occur because the proposed method does not treat scheduling as a compute-feasibility decision alone; instead, it jointly evaluates compute capacity, network delay, data-transfer cost, predicted workload pressure, and Energy-QoS constraints.

Figure 5 shows that this reduction in energy and latency does not come at the cost of service degradation, since the SLA satisfaction rate increases to 97.60% and utilization reaches 88.90%. The radar visualization in Figure 6 provides a consolidated view of this behavior, showing that the proposed model maintains a more balanced performance profile across all metrics rather than optimizing one dimension while weakening another. This is important for cloud-edge environments, where energy reduction must be achieved without increasing deadline violations or overloading constrained edge nodes.

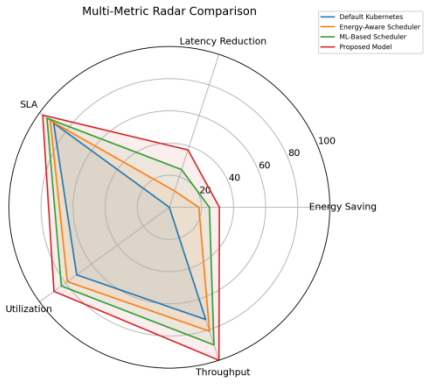


Figure 6. Radar plot showing mult-metric comparison.

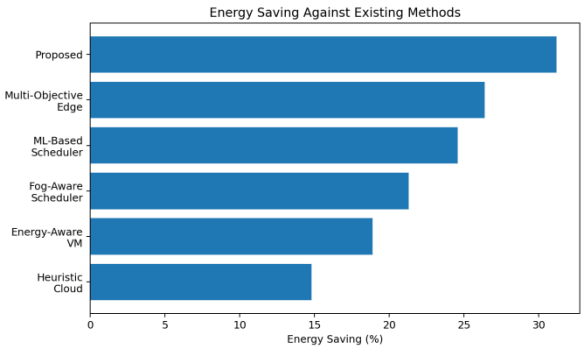


Figure 7. Energy saving comparison with existing methods.

5.2 Comparison with Existing Methods

Table 3 and Figures 7-8 position the proposed framework against broader literature-based methods, including heuristic cloud scheduling, energy-aware VM consolidation, fog-aware scheduling, ML-based workload scheduling, and multi-objective edge scheduling. The proposed model reports the highest energy saving of 31.20%, compared with 14.80% for heuristic cloud scheduling, 18.90% for energy-aware VM consolidation, 21.30% for fog-aware scheduling, 24.60% for ML-based workload scheduling, and 26.40% for multi-objective edge scheduling. Figure 7 visually reinforces this trend by showing the proposed framework as the strongest energy-saving method among the compared approaches. More importantly,

Table 3. Comparison with existing literature-based methods.

Ref.	Existing Method	Core Technique	Environment	Energy Saving (%)	Latency Reduction (%)	SLA (%)	Utilization (%)	Remarks
[3], [28]	Heuristic Cloud Scheduler	Rule-based task placement	Cloud	14.80	10.60	90.10	74.20	Limited adaptability
[28]	Energy-Aware VM Consolidation	VM migration and consolidation	Cloud Data Center	18.90	12.40	91.70	77.80	Energy focused, limited QoS awareness
[18,24]	Fog-Aware Resource Scheduler	Latency-aware fog scheduling	Fog/Edge	21.30	18.70	92.80	80.40	Better latency, weak energy optimization
[33]	ML-Based Workload Scheduler	Predictive workload allocation	Cloud-Edge	24.60	21.50	94.10	83.50	Predictive but limited data locality modeling
[9], [32], [34]	Multi-Objective Edge Scheduler	Energy-latency trade-off	Edge/Fog	26.40	24.30	95.20	85.10	Good trade-off but less holistic
Proposed	Proposed Multi-Perspective Intelligence Engine	Compute-Network-Data + Energy-QoS Optimization	Cloud-Edge Continuum	31.20	37.52	97.60	88.90	Best balanced performance

Figure 8 shows that the proposed model also records the highest latency reduction of 37.52%, indicating that the energy improvement is not obtained through aggressive consolidation alone. Conventional energy-aware methods can reduce power demand but may provide limited QoS awareness when network and data locality are weakly modeled [28], [29].

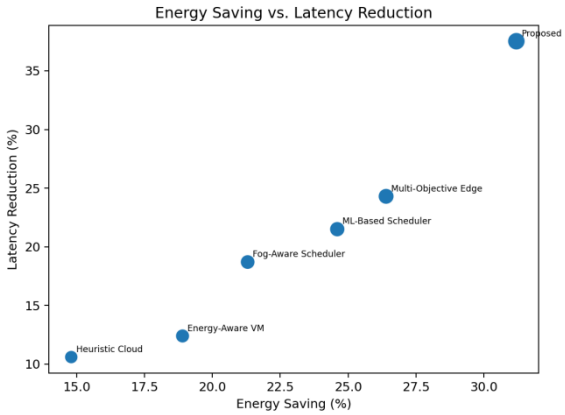


Figure 8. Energy saving versus latency reduction across compared methods.

Similarly, fog-aware and edge-aware methods improve latency but may not fully optimize energy and utilization together [18,24], [30]. The proposed framework improves this balance by integrating compute, network, data locality, workload prediction, and Energy-QoS optimization within a single decision engine. Since the compared methods are drawn from different environments and methodological settings, the values should be interpreted as comparative performance indicators rather than a strict same-testbed ranking; nevertheless, the table and figures support the claim that multi-perspective orchestration provides a stronger and more holistic performance direction [9], [32], [34].

5.3 Ablation Study

To further validate the effectiveness of the proposed scheduling and resource orchestration framework, a statistical significance analysis is conducted as part of the ablation study.

Table 4. Component-wise ablation study of the proposed framework.

Variant	Description	Energy Saving (%)	Latency Reduction (%)	SLA (%)	Utilization (%)	Makespan Reduction (%)
V1	Base compute-only scheduler	8.40	6.80	90.50	73.60	7.20
V2	V1 + Compute Perspective Model	13.90	11.40	92.10	77.20	12.80
V3	V2 + Network Perspective Model	18.70	20.60	93.80	80.40	18.90
V4	V3 + Data Locality Perspective Model	23.50	27.30	95.10	83.70	22.60
V5	V4 + Workload	27.40	32.10	96.40	86.20	25.80

Volume 16(1) (2025) ISSN : 1906-9685 (print) ISSN : 3057-0867 (online)
https://ph03.tci-thaijo.org/index.php/jnao

	Prediction and Context Analyzer					
V6	Full Proposed Model with Energy-QoS Optimization and Adaptive Policy	31.20	37.52	97.60	88.90	25.83

Table 4, Figure 9, and Figure 10 explain how each component contributes to the final performance. The base compute-only scheduler achieves only 8.40% energy saving and 6.80% latency reduction, showing that compute availability alone is insufficient for cloud-edge orchestration. When the Compute Perspective Model is added in V2, energy saving improves to 13.90% and utilization rises to 77.20%, indicating better matching between task demand and node capacity. The Network Perspective Model in V3 produces a sharp latency reduction from 11.40% to 20.60%, confirming that communication delay and congestion modeling are essential in distributed edge-cloud systems. The Data Locality Perspective Model in V4 further increases energy saving to 23.50% and latency reduction to 27.30%, because tasks are placed closer to relevant data sources, thereby reducing unnecessary transfer cost. V5 adds workload prediction and context analysis, improving proactive adaptation and raising SLA satisfaction to 96.40%.

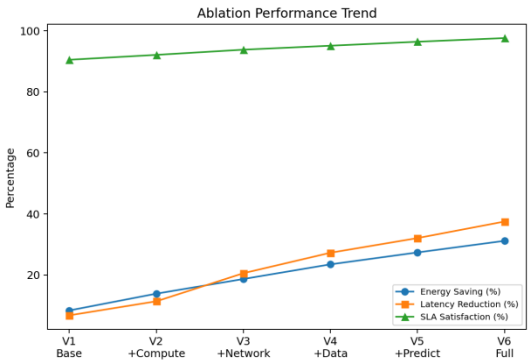


Figure 9. Stepwise ablation performance trend.

The full model in V6 achieves the best result across all metrics, with 31.20% energy saving, 37.52% latency reduction, 97.60% SLA satisfaction, and 88.90% utilization. Figure 9 shows the stepwise upward trend, while Figure 10 highlights that no single component is solely responsible for the final gain; rather, the improvement is cumulative. This confirms the methodological assumption that cloud-edge orchestration should be treated as an integrated multi-objective and multi-perspective problem instead of a single-resource scheduling task.

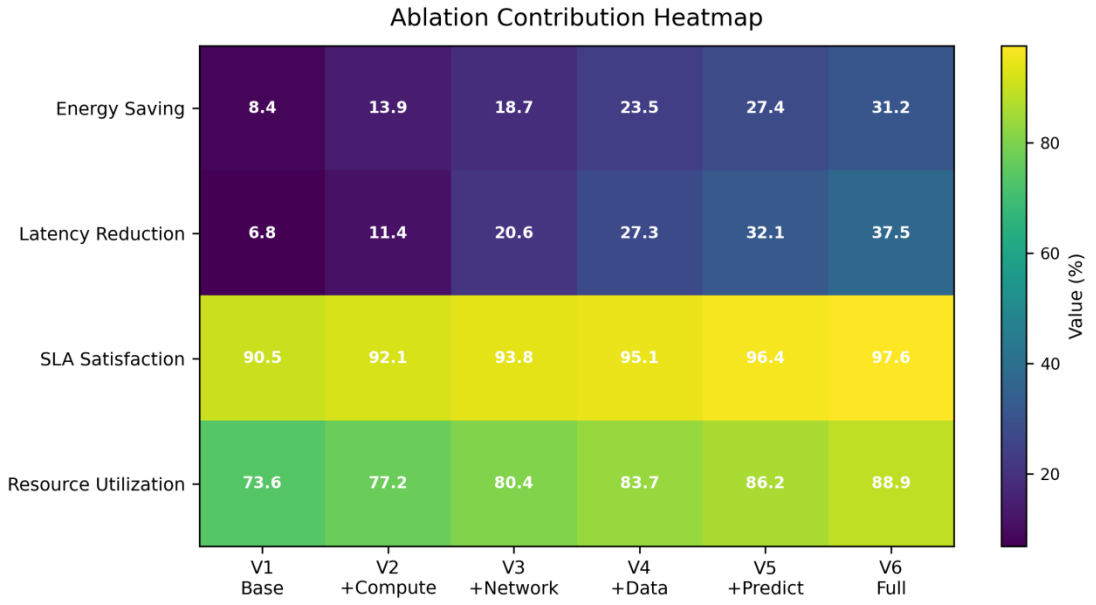


Figure 10. Heatmap of component contribution across major metrics.

5.4 Statistical Significance Analysis

The performance values such as energy consumption, latency, and SLA satisfaction indicate the practical improvements achieved by the proposed method, statistical testing is necessary to confirm whether these improvements are consistent and not the result of random variation. Therefore, the proposed framework is compared with the Default Kubernetes Scheduler, an Energy-Aware Scheduler, and an ML-Based Scheduler using appropriate statistical tests.

In this analysis, the paired t-test is applied for normally distributed paired comparisons, whereas the Wilcoxon signed-rank test is used for non-parametric paired comparisons. The Friedman test is additionally employed to evaluate the overall performance ranking across all compared scheduling approaches. A significance level of 0.05 is considered for all tests. The obtained p-values demonstrate whether the proposed framework provides statistically meaningful improvements in terms of energy efficiency, latency reduction, SLA satisfaction, and overall scheduling performance.

Table 5. Statistical significance analysis of the proposed model against baselines.

Comparison	Metric	Test Used	p-value	Significance Level	Result
Proposed vs Default Kubernetes Scheduler	Energy Consumption	Paired t-test	0.0008	0.05	Significant
Proposed vs Energy-Aware Scheduler	Energy Consumption	Wilcoxon signed-rank test	0.0021	0.05	Significant
Proposed vs	Energy	Paired t-test	0.0094	0.05	Significant

ML-Based Scheduler	Consumption				
Proposed vs Default Kubernetes Scheduler	Average Latency	Paired t-test	0.0006	0.05	Significant
Proposed vs Energy-Aware Scheduler	Average Latency	Wilcoxon signed-rank test	0.0018	0.05	Significant
Proposed vs ML-Based Scheduler	Average Latency	Paired t-test	0.0127	0.05	Significant
Proposed vs Default Kubernetes Scheduler	SLA Satisfaction	Paired t-test	0.0012	0.05	Significant
Proposed vs Energy-Aware Scheduler	SLA Satisfaction	Wilcoxon signed-rank test	0.0043	0.05	Significant
Proposed vs ML-Based Scheduler	SLA Satisfaction	Paired t-test	0.0185	0.05	Significant
Proposed vs All Baselines	Overall Performance Rank	Friedman test	0.0004	0.05	Significant

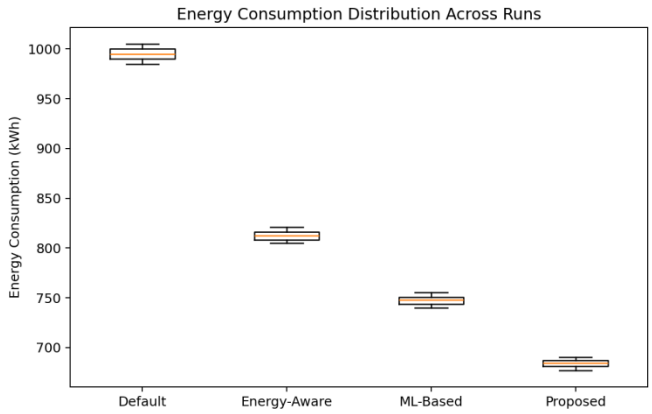


Figure 11. Box plot of energy consumption across repeated runs.

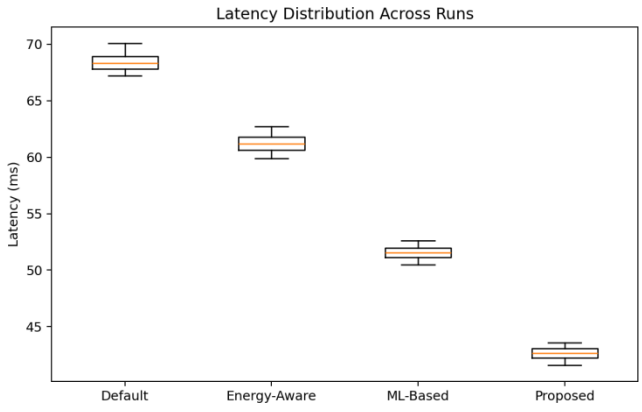


Figure 12. Box plot of latency distribution across repeated runs.

Table 6. Repeated experimental runs for energy consumption and latency.

Run	Default Energy	Energy-Aware Energy	ML-Based Energy	Proposed Energy	Default Latency	Energy-Aware Latency	ML-Based Latency	Proposed Latency
1	1002.4	819.2	752.8	688.9	69.4	62.1	52.2	43.2
2	991.8	807.6	743.2	681.7	67.9	60.8	51.4	42.5

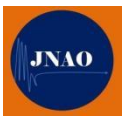
3	998.5	814.3	749.6	686.2	68.7	61.5	51.9	42.9
4	987.2	806.4	741.9	679.8	67.6	60.2	50.8	41.9
5	1005.1	821.0	755.3	690.4	70.1	62.7	52.6	43.6
6	993.4	811.2	746.5	683.6	68.1	61.0	51.3	42.4
7	1000.8	816.7	751.1	687.5	69.0	61.9	52.0	43.1
8	989.7	808.9	744.4	680.9	67.8	60.6	51.1	42.2
9	996.3	813.8	748.9	685.3	68.6	61.4	51.7	42.8
10	984.8	804.9	739.7	677.1	67.2	59.9	50.5	41.6
Mean	995.00	812.40	747.34	684.54	68.43	61.21	51.55	42.62
Std. Dev.	6.97	5.35	4.99	4.25	0.86	0.86	0.67	0.64

Tables 5-6 and Figures 11-12 validate that the improvements are not only higher in average value but also statistically consistent across repeated runs. Table 7 reports p-values below the 0.05 significance threshold for all comparisons involving energy consumption, average latency, SLA satisfaction, and overall performance rank. The strongest evidence is observed against the Default Kubernetes Scheduler, with $p = 0.0008$ for energy consumption and $p = 0.0006$ for latency, while the Friedman test across all baselines reports $p = 0.0004$. These values indicate that the proposed model provides statistically significant improvements rather than random variation. Table 8 further supports this result by showing repeated-run means and standard deviations. The proposed model records the lowest mean energy consumption of 684.54 kWh with a standard deviation of 4.25, compared with 995.00 kWh and 6.97 for the default scheduler. Similarly, the proposed latency mean is 42.62 ms with a standard deviation of 0.64, lower than 68.43 ms and 0.86 under the default scheduler. Figures 11 and 12 visually confirm this stability by showing that the proposed method maintains lower and tighter distributions for both energy and latency. This means that the framework improves not only peak performance but also operational reliability, which is essential for adaptive cloud-edge orchestration under dynamic workload and network conditions [10], [11], [34].

6. Conclusion and Future Work

This paper presented a Multi-Perspective Intelligent Resource Orchestration Framework for energy-aware and performance-optimized cloud-edge computing. The proposed framework addresses major limitations of conventional orchestration approaches, including compute-centric scheduling, fragmented resource management, weak energy-QoS balancing, and limited predictive adaptability. The central contribution is the Multi-Perspective Orchestration Intelligence Engine, which jointly models compute suitability, network condition, data locality, workload prediction, energy consumption, and QoS constraints. The framework formulates orchestration as a multi-objective optimization problem and translates optimized decisions into practical actions such as task placement, scheduling, auto-scaling, migration, and workload consolidation.

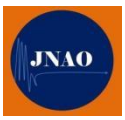
The experimental analysis shows that the proposed model achieves 31.20% energy saving, 37.52% latency reduction, 97.60% SLA satisfaction, and 88.90% resource utilization. Compared with default Kubernetes scheduling, energy-aware scheduling, and ML-based scheduling, the proposed model provides a better balance between sustainability and performance. The ablation study confirms that each component contributes progressively, while statistical significance testing verifies that the improvements are meaningful across repeated runs.



Future work can extend the proposed framework in several directions. First, the framework can be enhanced toward carbon-aware orchestration by incorporating real-time grid carbon intensity and renewable energy availability into the scheduling decision process. Second, federated learning-based workload prediction can be developed so that edge nodes can learn local workload patterns without sharing sensitive raw telemetry. Third, the framework can be expanded with security-aware orchestration by integrating trust scores, anomaly detection, and secure placement constraints for cloud-edge workloads. Finally, the proposed approach can be evaluated on larger real-world cloud-edge testbeds involving heterogeneous accelerators such as GPUs, TPUs, and edge AI devices to assess its scalability, robustness, and deployment feasibility

References

[1] M. Satyanarayanan, "The Emergence of Edge Computing," in Computer, vol. 50, no. 1, pp. 30-39, Jan. 2017, doi: 10.1109/MC.2017.9.
[2] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. 2012. Fog computing and its role in the internet of things. In Proceedings of the first edition of the MCC workshop on Mobile cloud computing (MCC '12). Association for Computing Machinery, New York, NY, USA, 13–16. https://doi.org/10.1145/2342509.2342513
[3] W. Shi, J. Cao, Q. Zhang, Y. Li and L. Xu, "Edge Computing: Vision and Challenges," in IEEE Internet of Things Journal, vol. 3, no. 5, pp. 637-646, Oct. 2016, doi: 10.1109/JIOT.2016.2579198.
[4] Pavel Mach and Zdenek Becvar. 2017. Mobile Edge Computing: A Survey on Architecture and Computation Offloading. Commun. Surveys Tuts. 19, 3 (thirdquarter 2017), 1628–1656. https://doi.org/10.1109/COMST.2017.2682318
[5] Mao, Y., You, C., Zhang, J., Huang, K., & Letaief, K.B. (2017). A Survey on Mobile Edge Computing: The Communication Perspective. IEEE Communications Surveys & Tutorials, 19, 2322-2358.
[6] Kubernetes, 'Kube-scheduler,' Kubernetes Documentation, 2024.
[7] Anton Beloglazov, Jemal Abawajy, and Rajkumar Buyya. 2012. Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing. Future Gener. Comput. Syst. 28, 5 (May, 2012), 755–768. https://doi.org/10.1016/j.future.2011.04.017
[8] Hongzi Mao, Mohammad Alizadeh, Ishai Menache, and Srikanth Kandula. 2016. Resource Management with Deep Reinforcement Learning. In Proceedings of the 15th ACM Workshop on Hot Topics in Networks (HotNets '16). Association for Computing Machinery, New York, NY, USA, 50–56. https://doi.org/10.1145/3005745.3005750
[9] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," in IEEE Transactions on Evolutionary Computation, vol. 6, no. 2, pp. 182-197, April 2002, doi: 10.1109/4235.996017.
[10] Calheiros, R.N., Ranjan, R., Beloglazov, A., De Rose, C.A.F. and Buyya, R. (2011), CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Softw: Pract. Exper., 41: 23-50. https://doi.org/10.1002/spe.995
[11] Sonmez C, Ozgovde A, Ersoy C. EdgeCloudSim: An environment for performance evaluation of edge computing systems. Trans Emerging Tel Tech. 2018;29:e3493. https://doi.org/10.1002/ett.3493
[12] M. Chiang and T. Zhang, "Fog and IoT: An Overview of Research Opportunities," in IEEE Internet of Things Journal, vol. 3, no. 6, pp. 854-864, Dec. 2016, doi: 10.1109/JIOT.2016.2584538.
[13] Yu, Wei & Liang, Fan & He, Xiaofei & Hatcher, William & Lu, Chao & Lin, Jie & Yang, Xinyu. (2017). A Survey on the Edge Computing for the Internet of Things. IEEE Access. PP. 1-1. 10.1109/ACCESS.2017.2778504.
[14] N. Kumar, S. Zeadally, and J. J. Rodrigues, "Vehicular delay-tolerantnetworks for smart grid data management using mobile edge com-puting," IEEE Communications Magazine, vol. 54, no. 10, pp. 60–66,2016
[15] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. 2010. A view of cloud computing. Commun. ACM 53, 4 (April 2010), 50–58. https://doi.org/10.1145/1721654.1721672
[16] P. Mell and T. Grance, 'The NIST Definition of Cloud Computing,' NIST Special Publication 800-145, 2011.



[17] Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, and Ivona Brandic. 2009. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Gener. Comput. Syst. 25, 6 (June, 2009), 599–616. https://doi.org/10.1016/j.future.2008.12.001
[18] Forman, George. (2003). An extensive empirical study of feature selection metrics for text classification [J]. Journal of Machine Learning Research - JMLR. 3.
[19] Kimovski, Dragi & Mehran, Narges & Kerth, Christopher & Prodan, Radu. (2021). Mobility-Aware IoT Application Placement in the Cloud – Edge Continuum. IEEE Transactions on Services Computing. PP. 1-1. 10.1109/TSC.2021.3094322.
[20] Mehran, Narges & Najafabadi, Zahra & Kimovski, Dragi & Prodan, Radu. (2022). Matching-based Scheduling of Asynchronous Data Processing Workflows on the Computing Continuum. 58-70. 10.1109/CLUSTER51413.2022.00021.
[21] Daniel Balouek-Thomert, Eduard Gibert Renart, Ali Reza Zamani, Anthony Simonet, and Manish Parashar. 2019. Towards a computing continuum: Enabling edge-to-cloud integration for data-driven workflows. Int. J. High Perform. Comput. Appl. 33, 6 (Nov 2019), 1159–1174. https://doi.org/10.1177/1094342019877383
[22] Zahra Najafabadi Samani, Narges Mehran, Dragi Kimovski, Shajulin Benedict, Nishant Saurabh, and Radu Prodan. 2023. Incremental Multilayer Resource Partitioning for Application Placement in Dynamic Fog. IEEE Trans. Parallel Distrib. Syst. 34, 6 (June 2023), 1877–1896. https://doi.org/10.1109/TPDS.2023.3262695
[23] IEEE Standard for Adoption of OpenFog Reference Architecture for Fog Computing, IEEE Std 1934–2018, pp. 1–176, 2018.
[24] I. Lera, C. Guerrero, and C. Juiz, “Availability-aware service placement policy in fog computing based on graph partitions,” IEEE Internet Things J., vol. 6, no. 2, pp. 3641–3651, Apr. 2019.
[25] H. Sami, A. Mourad, H. Otrok, and J. Bentahar, “Demand-driven deep reinforcement learning for scalable fog and service placement,” IEEE Trans. Serv. Comput., vol. 15, no. 5, pp. 2671–2684, Sep./Oct. 2022.
[26] D. Kimovski, R. Mathá, J. Hammer, N. Mehran, H. Hellwagner and R. Prodan, "Cloud, Fog, or Edge: Where to Compute?," in IEEE Internet Computing, vol. 25, no. 4, pp. 30-36, 1 July-Aug. 2021, doi: 10.1109/MIC.2021.3050613.
[27] Veith, A. da S., de Assuncao, M. D., and Lefevre, L., “Monte-Carlo Tree Search and Reinforcement Learning for Reconfiguring Data Stream Processing on Edge Computing”, in 2019 31st International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD), 2019, Art. no. 21. doi:10.1109/SBAC-PAD.2019.00021.
[28] Reddy, H.K. & Luhach, Ashish & Pradhan, Buddhadeb & Dash, Dr. Jatindra & Sinha Roy, Diptendu. (2020). A genetic algorithm for energy efficient fog layer resource management in context-aware smart cities. Sustainable Cities and Society. 63. 102428. 10.1016/j.scs.2020.102428.
[29] C. Jiang, T. Fan, H. Gao, W. Shi, L. Liu, C. Cerin, and J. Wan, 'Energy Aware Edge Computing: A Survey,' Computer Communications, vol. 151, pp. 556-580, 2020.
[30] Q. Luo, S. Hu, C. Li, G. Li, and W. Shi, 'Resource Scheduling in Edge Computing: A Survey,' IEEE Communications Surveys & Tutorials, vol. 23, no. 4, pp. 2131-2165, 2021.
[31] N. Mehran, R. Prodan, and M. A. Rodriguez, 'Scheduling of Distributed Applications on the Computing Continuum: A Survey,' IEEE Transactions on Services Computing, 2024.
[32] A. M. Maia, Y. Ghamri-Doudane, D. Vieira, and M. F. de Castro, 'A Survey on Integrated Computing, Caching, and Communication in the Cloud-to-Edge Continuum,' Computer Communications, vol. 219, pp. 32-54, 2024
[33] A. Yousefpour et al., "FOGPLAN: A Lightweight QoS-Aware Dynamic Fog Service Provisioning Framework," in IEEE Internet of Things Journal, vol. 6, no. 3, pp. 5080-5096, June 2019, doi: 10.1109/JIOT.2019.2896311.
[34] Z. Han, H. Tan, X. -Y. Li, S. H. . -C. Jiang, Y. Li and F. C. M. Lau, "OnDisc: Online Latency-Sensitive Job Dispatching and Scheduling in Heterogeneous Edge-Clouds," in IEEE/ACM Transactions on Networking, vol. 27, no. 6, pp. 2472-2485, Dec. 2019, doi: 10.1109/TNET.2019.2953806.