Computer Modeling in Engineering & Sciences

Tech Science Press

Doi:10.32604/cmes.2025.060973

ARTICLE

Quantum Inspired Adaptive Resource Management Algorithm for Scalable and Energy Efficient Fog Computing in Internet of Things (IoT)

Sonia Khan[1], Naqash Younas[2], Musaed Alhussein[3], Wahib Jamal Khan[2], Muhammad Shahid Anwar[4,*] and Khursheed Aurangzeb[3]

[1]Youth Office Haripur, Directorate of Youth Affairs, Government of Khyber Pakhtunkhwa, Haripur, 22620, Pakistan
[2]School of Software Engineering, Dalian University of Technology, Ganjingzi District, Dalian, 116024, China
[3]Department of Computer Engineering, College of Computer and Information Sciences, King Saud University, Riyadh, 11543, Saudi Arabia
[4]Department of AI and Software, Gachon University, Seongnam-si, 13120, Republic of Korea

*Corresponding Author: Muhammad Shahid Anwar. Email: shahidanwar786@gachon.ac.kr

Received: 13 November 2024; Accepted: 10 February 2025; Published: 03 March 2025

ABSTRACT: Effective resource management in the Internet of Things and fog computing is essential for efficient and scalable networks. However, existing methods often fail in dynamic and high-demand environments, leading to resource bottlenecks and increased energy consumption. This study aims to address these limitations by proposing the Quantum Inspired Adaptive Resource Management (QIARM) model, which introduces novel algorithms inspired by quantum principles for enhanced resource allocation. QIARM employs a quantum superposition-inspired technique for multi-state resource representation and an adaptive learning component to adjust resources in real time dynamically. In addition, an energy-aware scheduling module minimizes power consumption by selecting optimal configurations based on energy metrics. The simulation was carried out in a 360-minute environment with eight distinct scenarios. This study introduces a novel quantum-inspired resource management framework that achieves up to 98% task offload success and reduces energy consumption by 20%, addressing critical challenges of scalability and efficiency in dynamic fog computing environments.

KEYWORDS: Quantum computing; resource management; energy efficiency; fog computing; Internet of Things

1 Introduction

The exponential growth of the Internet of Things (IoT) has revolutionized the digital domain, allowing interconnected devices to share and process data with unprecedented efficiency [1,2]. Along with the growth in IoT devices, the demand for powerful and decentralized data processing solutions has led to the adoption of fog computing [3,4]. Unlike traditional cloud computing [5], it operates at the edge of a network [6], providing localized processing, storage, and control services to reduce latency and therefore improve real-time response [7,8]. This paradigm faces many challenges because, in general, fog nodes are resource-restricted compared to centralized cloud servers [9,10]. Most existing methods suffer from a lack of scalability [11], as the dense volume and diversity of IoT devices can rapidly overwhelm processing resources [12]. In addition, most models include machine learning and deep learning algorithms that are computationally intensive and thus may impede real-time responsiveness [13,14]. Energy efficiency is another major concern because most existing models do not balance processing needs with power limitations of edge devices [15]. With these challenges, the need for innovation in resource management techniques that can

dynamically adapt to real-time demand is felt without the expense of unnecessary computational and energy overhead [16,17].

This work is motivated by the development of a framework for resource management to address the limitations found in existing approaches. Although useful, current methods often suffer from a general lack of flexibility and efficiency in providing scalable and sustainable IoT fog ecosystems [18]. This research aims to fill this gap by proposing a new quantum-inspired resource management algorithm [19], able to work within the limit of limited computing resources and energy availability. The central problem lies in achieving adaptive, energy-efficient resource allocation that aligns with the dynamic demands of real-time IoT applications [20]. Therefore, this study poses the following research question: How can a quantum-inspired adaptive resource management (QIARM) algorithm enhance scalability, energy efficiency, and adaptability in fog computing environments?

The proposed approach integrates principles from quantum computing, particularly quantum superposition, and entanglement for resource management in fog computing, as shown in Fig. 1. The proposed algorithm incorporates a quantum-inspired data structure that enables the simultaneous representation of multiple states of resource allocation. By simulating superposition, QIARM parallelly explores possible allocations and is thus much faster in its decisions. In addition, it proposes an entanglement-inspired mechanism to enable effective resource coordination, which simultaneously synchronizes the resource status of distributed fog nodes to mitigate redundant allocation and latency issues. In addition to this, QIARM includes an adaptive learning part that adjusts the strategies for allocation at runtime, according to network dynamics, and strikes a proper trade-off between performance and energy efficiency. Then, the integrated energy-aware scheduling model will ensure minimum consumption of power by favoring low-energy resource states with no sacrifice of computational effectiveness. The main contributions of this research study can be summarized as follows:

- The proposed algorithm introduces a novel data structure inspired by quantum superposition that enables simultaneous consideration of multiple resource states.
- A mechanism synchronizes allocation across the nodes and is very effective in terms of not allowing redundancy of resources, which in turn has a positive impact on the latency performance of a multi-node fog system. The approach also builds on an entanglement-inspired model for coordinating resource management across distributed fog nodes.
- QIARM integrates a unique and energy-conscious scheduling component that continuously monitors resource demand and adapts allocation strategies to minimize power consumption. This feature is particularly beneficial for IoT-fog environments where energy resources are limited, providing sustainable operational efficiency.

The structure of this article can be summarized as follows: Section 2 reviews existing approaches to resource management to identify relevant research gaps. Section 3 introduces the Quantum Inspired Adaptive Resource Management (QIARM) approach, detailing its core components and methodology. Section 4 presents the experimental setup and simulation results to evaluate the QIARM performance. Section 6 provides a discussion of the results and their implications. Finally, Section 7 concludes the article.

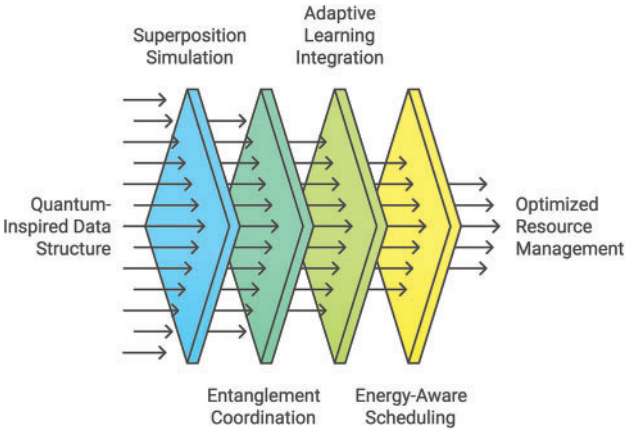

Figure 1: The overview of the proposed QIARM approach

2 Related Work

The rapid proliferation of IoT devices has significantly increased the demand for resource-efficient computing and real-time data processing capabilities. Fog computing, which allows computation at the edge of the network, has emerged as a solution to address latency and bandwidth constraints in IoT environments [21,22]. Recent research has focused on developing efficient and adaptive resource management strategies to optimize performance and ensure responsiveness under varying conditions [23,24]. In fog computing environments, task offloading and resource allocation have become central to reducing delays and improving the quality of service for IoT applications [25,26]. Advanced resource management techniques have been proposed to address specific requirements, such as network splitting in 5G and 6G systems [27,28], retransmission strategies in the Long Range Wide Area Network (LoRaWAN) [29], and adaptive management of greenhouse environments [30].

Various approaches to resource management leverage machine learning and deep learning techniques to handle the complexities of IoT networks. For example, lightweight frameworks for distributed computing aim to balance computational load across IoT nodes, effectively managing resource constraints [31]. In addition to classifier systems, fog-cloud hybrid models have also been introduced to improve resource utilization by making intelligent decisions in distributed environments [32]. In Industrial IoT, dual-driven models have been presented based on reinforcement learning to improve sustainable computing [33]. Digital twin and blockchain-supported environments have also been suggested [34,35]. Furthermore, reinforcement learning has been used to enable resource management for deep neural network inference and improve computational efficiency in IoT networks where real-time analytics plays a key role [36].

Novel adaptive frameworks in IoT are still emerging that will include predictive maintenance and prioritization of resources to improve the reliability of the service and reduce any interruption to it [37,38]. This has been achieved in resource management on mobile edge computing using a priority mechanism to optimize resource allocation and achieve low latency for mobile users [26]. Pervasive edge computing models have also been popular, especially for applications that require very high data transmission rates, including VR streaming wirelessly in industrial IoT applications [39]. However, it has yet to be overcome concerning developing a scalable, adaptable, and energy-efficient resource management framework that can cope with the fluctuating demands of real-time IoT applications.

Another key limitation of existing resource management strategies is that they cannot respond appropriately to rapid changes in network load and availability, as reflected in predictive maintenance

and priority-based models for mobile edge environments [26,37]. These models also do not consider any coordination mechanisms required by applications for sharing resources at different nodes, which is typical of latency-sensitive applications such as virtual reality streaming in industrial IoT [39]. The proposed QIARM approach aims to address challenges by introducing quantum-inspired resource representation and entanglement-like coordination mechanisms for runtime adjustments of resources. This allows QIARM to represent multiple states of allocation in superposition, leading to searching larger solution spaces while reducing computational overhead and deploying faster resource allocations. Furthermore, the energy-aware scheduling of QIARM optimizes energy consumption by adapting the allocation according to run-time conditions, which is not supported by state-of-the-art models. Fig. 2 summarizes the key limitations in existing methods and underlines how the proposed approach is addressing these challenges.

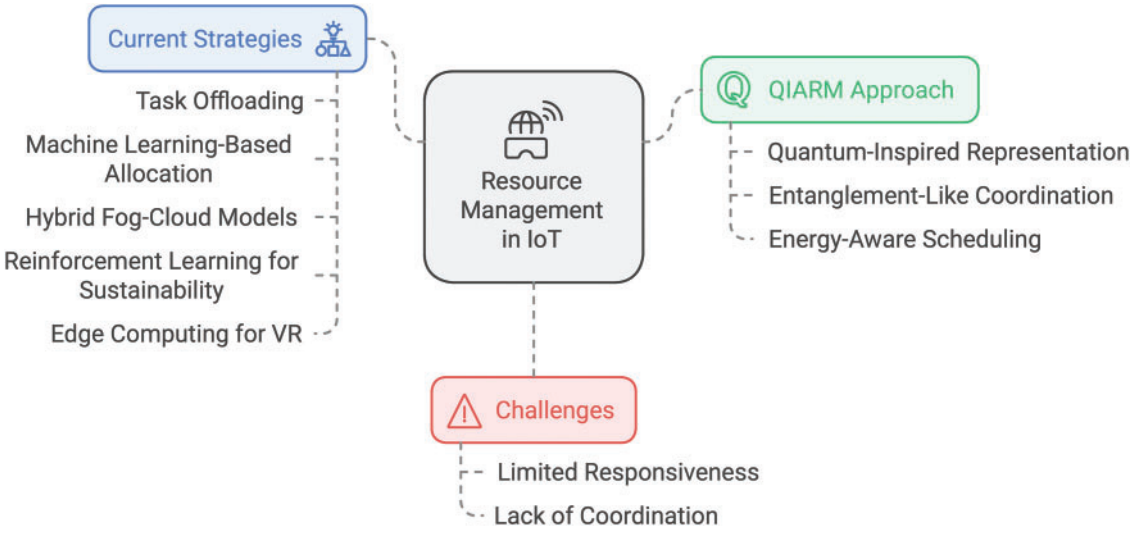

Figure 2: Comparison of the existing approaches limitations, challenges, and QIARM solutions

3 Proposed QIARM Approach

Quantum-inspired adaptive Resource Management switches on quite a new conceptual framework toward optimized resource allocation in fog computing environments. Featuring quantum-inspired superposition and entanglement, QIARM dynamically coordinates the distributed resources over time to make it adaptive with low latency and energy efficiency. This allows for addressing scalability challenges and enables real-time responsiveness uniquely for IoT applications over resource-constrained networks.

3.1 Workflow of the Proposed QIARM Approach

The QIARM approach is designed for efficient and adaptive resource allocation in dynamic fog computing environments. Comprising three major phases, each is tailored for managing resources at different levels: initialization, real-time adaptation, and real-time adaptation forming a holistic and responsive framework. The workflow of the QIARM approach is represented in Fig. 3, which depicts an end-to-end process to initialize, allocate, and dynamically adapt resources across fog nodes.

3.1.1 Initialization Phase

The Initialization Phase sets up the QIARM framework by defining and configuring the resource states across all fog nodes. This is achieved by providing quantum-inspired representations of resources,

whereby each node initializes a state from the available computing, storage, and energy resources. Due to the superposition principles, multiple configurations of resources can be represented at once, forming a large set of possible states with no extra computation.

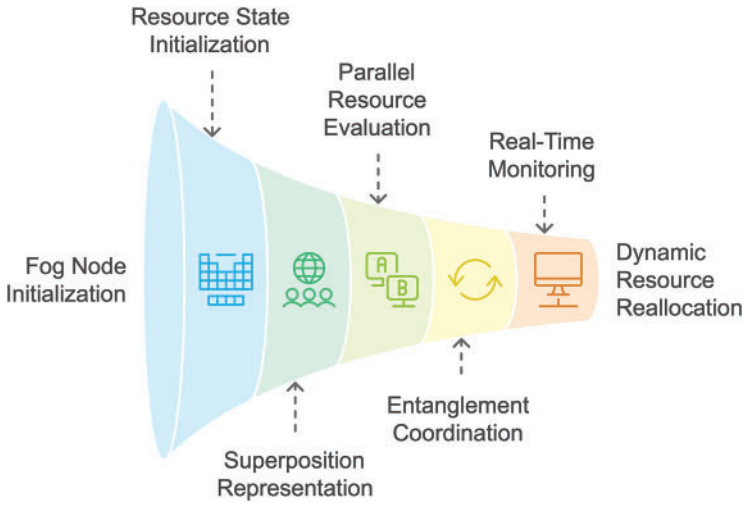

Figure 3: Workflow of the QIARM approach

3.1.2 Resource Allocation Phase

After initialization has taken place, the actual assignment of the resources is carried out by the resource allocation phase. The superposition and entanglement-inspired models enable consideration of resource states at nodes in a parallel process viewed to be emulating the simultaneous exploration of many options regarding the allocation of resources. Therefore, superposition allows the system to analyze in parallel several possible allocations to establish those configurations that will yield maximum efficiency. Entanglement-inspired mechanisms are used to coordinate resources across nodes, ensuring that distributed resources are indeed effectively synchronized.

3.1.3 Adaptive Adjustment Phase

The Adaptive Adjustment Phase introduces a continuous monitoring mechanism based on feedback that determines the adaptation in real-time concerning the resource requirements that are constantly changing. Monitoring in this phase will include tracking network conditions, usage patterns, and computational loads by adjusting resource allocation in real-time to receive real-time feedback. Upon detecting shifted demands, QIARM's adaptive learning module will reassess the state of the resources and reconfigure them to meet existing needs on the network.

3.2 System Model

In the proposed QIARM system, the quantum-inspired approach is emphasized, uniquely aligning resource allocation strategies with real-time network conditions. In Algorithm 1, QIARM workflow is outlined, detailing the initialization of resource states, the entanglement-inspired resource allocation process, and the adaptive adjustments to maintain balanced load distribution across fog nodes in real-time.

Algorithm 1: QIARM Workflow for Fog Computing Environment

Input: Set of IoT devices $\{D_1, D_2, \ldots, D_N\}$ generating demands $d_i(t)$ with mean $\lambda_i(t)$, Set of fog nodes $\{F_1, F_2, \ldots, F_M\}$ with resource capacities (C_j, M_j, E_j)

Output: Optimal resource allocation across fog nodes to handle IoT demands

Initialization Phase:
foreach *fog node* $F_j(j = 1, \ldots, M)$ **do**
  Initialize resource vector $\mathbf{R}_{ij} = \left(C_{ij}, M_{ij}, E_{ij}\right)$ for each task i;
  Establish quantum-inspired superposition states for $\mathbf{R}_{ij}$
end
Resource Allocation Phase:
Compute the entanglement function $Q(F_i, F_j) = \exp\left(-\alpha \left|\mathbf{R}_i - \mathbf{R}_j\right|\right)$ between fog nodes **foreach** *task i from IoT device* D_i **do**
  foreach *fog node* F_j **do**
    Calculate resource demand $d_i(t)$ based on Poisson model with mean $\lambda_i(t)$ Compute availability $A_j(t)$ for each F_j as per energy constraint E_j and resource vector $\mathbf{R}_{ij}$ Determine adaptive matching $\mathcal{M}(d_i, A_j)$ for task i allocation to node j
  end
  Assign task i to the fog node j^* with the highest $\mathcal{M}(d_i, A_j)$;
end
Adaptive Adjustment Phase:
while *Resource demand or availability changes in real-time* **do**
  foreach task i **do**
    Update $\lambda_i(t)$ and $d_i(t)$ for IoT device D_i;
    Re-evaluate $A_j(t)$ for each fog node F_j;
    Adjust $\mathcal{M}(d_i, A_j)$ based on updated $d_i(t)$ and $A_j(t)$ to balance load;
    Redirect task i if a better F_j with a higher $\mathcal{M}(d_i, A_j)$ exists;
  end
end

3.2.1 Objective Function

The overarching goal of the proposed QIARM model is to optimize resource allocation by balancing latency, energy efficiency, and task offloading success rate. This is achieved by minimizing the objective function J, defined as:

$$J = w_1 \cdot L + w_2 \cdot E - w_3 \cdot S$$

Here, L, E, and S represent latency (ms), energy consumption (kWh) and task success rate (%), respectively. The weighting factors w_1, w_2, and w_3 allow prioritizing specific metrics, such that $w_1 + w_2 + w_3 = 1$. These weights can be tuned according to the application requirements to emphasize particular objectives, such as energy efficiency for green IoT systems or low latency for real-time applications.

3.2.2 Fog Computing Environment

The fog computing environment consists of a set of N IoT devices, $\{D_1, D_2, \ldots, D_N\}$, each generating variable workloads, and a set of M fog nodes, $\{F_1, F_2, \ldots, F_M\}$, capable of executing tasks with limited resources. Each fog node F_j $(j = 1, \ldots, M)$ has computational capacity C_j, memory capacity M_j, and energy

restriction E_j. To model resource types across fog nodes, let $\mathbf{R}_{ij}$ represent the resource vector for task i at fog node j, where

$$\mathbf{R}_{ij} = \left(C_{ij}, M_{ij}, E_{ij}\right), \tag{1}$$

with C_{ij}, M_{ij}, and E_{ij} denoting the computation, memory, and energy requirements, respectively, for each task i at node j. The interactions within the fog environment are captured through a quantum-inspired entanglement function, $\mathcal{Q}(F_i, F_j)$, that synchronizes resource management across nodes:

$$\mathcal{Q}(F_i, F_j) = \exp\left(-\alpha \left|\mathbf{R}_i - \mathbf{R}_j\right|\right), \tag{2}$$

where α is the scaling parameter that dictates the extent of entanglement concerning resource similarity between fog nodes. The function furthers the coordinated allocation through dynamic changes of weights with alterations to that encourage a distributed resource-sharing method that is efficient.

3.2.3 Resource Demand and Availability Modeling

The demand and supply model is adaptive, where real-time demand fluctuations are captured using the probabilistic framework. Let $d_i(t)$ denote the demand for resources from the IoT device D_i at the instant t modeled as Poisson random variable with mean $\lambda_i(t)$, where:

$$d_i(t) \sim \text{Poisson}(\lambda_i(t)), \tag{3}$$

and $\lambda_i(t)$ dynamically adjusts based on observed demand patterns. Resource availability for each fog node F_j is represented by an availability function $A_j(t)$, formulated as:

$$A_j(t) = C_j \cdot f(E_j, t) - \sum_{i=1}^{N} \mathbf{R}_{ij}, \tag{4}$$

where $f(E_j, t)$ is a time-dependent function representing the energy efficiency of node j hence ensuring that dynamically updated availability is matched with energy constraints over time. Demand is matched with availability using the adaptive matching function $\mathcal{M}(d_i, A_j)$ assigning tasks from IoT devices to fog nodes considering resource constraints and availability probabilities:

$$\mathcal{M}(d_i, A_j) = \frac{\mathcal{Q}(F_i, F_j) \cdot A_j(t)}{1 + e^{-\beta\left(d_i(t) - A_j(t)\right)}}, \tag{5}$$

where β is an adaptation parameter that controls the sensitivity of task assignments based on real-time demand and availability.

3.3 Quantum-Inspired Algorithm Design

QIARM is proposed to improve resource management for fog computing environments by the principles of quantum mechanics. In the following, the proposed framework provides a representation based on the quantum superposition of resources, an entanglement-inspired mechanism for coordination among the nodes, an adaptive learning feature, and an energy-sensitive scheduling system.

3.3.1 Resource Representation via Quantum Superposition

In the QIARM model, resource allocation states are represented using a quantum superposition-inspired approach to enable parallel exploration of multiple configurations. Each fog node F_j is assigned

a quantum state vector Ψ_j, which encapsulates all possible allocations simultaneously, allowing for a probabilistic assessment of each configuration.

Define Ψ_j as a superposition of the potential resource states $|\phi_k\rangle$, where each $|\phi_k\rangle$ corresponds to a different configuration of the allocation for node j:

$$\Psi_j = \sum_{k=1}^{K} \alpha_{jk} |\phi_k\rangle, \tag{6}$$

where α_{jk} is the amplitude of state $|\phi_k\rangle$ for node j, representing the likelihood of selecting configuration k. The values of α_{jk} are normalized such that:

$$\sum_{k=1}^{K} |\alpha_{jk}|^2 = 1. \tag{7}$$

Each state $|\phi_k\rangle$ in Ψ_j encodes a vector of resource parameters (C_k, M_k, E_k), where C_k is the computation capacity, M_k is the memory, and E_k is the energy required for configuration k. We can consider any one of the resource parameters, say for node j, by taking an expectation value weighted across all configurations:

$$\langle C_j \rangle = \sum_{k=1}^{K} |\alpha_{jk}|^2 C_k, \quad \langle M_j \rangle = \sum_{k=1}^{K} |\alpha_{jk}|^2 M_k, \quad \langle E_j \rangle = \sum_{k=1}^{K} |\alpha_{jk}|^2 E_k. \tag{8}$$

The quantum-inspired superposition technique enables the system to assess all resource states in parallel and picks up the best allocation from the measurement of Ψ_j. The final measurement collapses Ψ_j to $|\phi_{k^*}\rangle$ that represents the best allocation with highest probability:

$$k^* = \arg\max_k |\alpha_{jk}|^2. \tag{9}$$

This parallel exploration provides a novel and efficient method to handle resource allocation in fog nodes, where traditional sequential evaluations would be computationally prohibitive.

3.3.2 Entanglement-Inspired Coordination Mechanism

QIARM follows the principles of quantum entanglement for synchronizing resource allocations across distributed fog nodes. This ensures efficiency in resource management in a nonredundant manner. Entanglement-inspired coordination dynamically models node interdependencies, promoting shared resource utilization and helping nodes balance resource allocation against dynamic demands in real-time.

Let Φ_{ij} be the entanglement coefficient between two fog nodes, F_i and F_j. It measures the strength of their resource correlation. The higher the value is, the stronger the resource-sharing potential will be. Now, define Φ_{ij} as:

$$\Phi_{ij} = \exp\left(-\gamma \left\| \mathbf{R}_i - \mathbf{R}_j \right\| \right), \tag{10}$$

where γ is a scaling factor, while $\mathbf{R}_i$ and $\mathbf{R}_j$ are the resource vectors for nodes F_i and F_j, respectively, and the entanglement coefficient Φ_{ij} decreases due to the increasing difference between resource vectors, thus reducing the intensity of coordination for diverging resource states.

For correlated allocation, define the vector of the shared resource, $\mathbf{R}_{\text{shared}}$, with components weighted by contributions of the entangled nodes:

$$\mathbf{R}_{\text{shared}} = \frac{\sum_{i \neq j} \Phi_{ij} \mathbf{R}_i}{\sum_{i \neq j} \Phi_{ij}}, \tag{11}$$

The shared vector $\mathbf{R}_{\text{shared}}$ is the aggregated resource state of entangled nodes; this, in effect, aligns resource availability within the fog environment. Each node, finally, uses $\mathbf{R}_{\text{shared}}$ for its local computation to harmonize its resource distribution.

For task allocation, let us define a coordination function $C_{ij}(t)$, which in real-time and based on mutually perceived availability and demand for each other, readjusts resource distribution between entangled nodes. The function may be expressed as:

$$C_{ij}(t) = \frac{\Phi_{ij} \cdot \left(A_i(t) + A_j(t)\right)}{1 + e^{-\delta\left(d_i(t) - d_j(t)\right)}}, \tag{12}$$

where $A_i(t), A_j(t)$ are the availability functions of nodes F_i, F_j, respectively, $d_i(t)$ and $d_j(t)$ their respective demands, δ is a sensitivity parameter modulating response to the real-time demand disparity. Finally, the resource allocation decision for each node incorporates both local as well as shared resources given by:

$$\mathbf{R}_i^{\text{final}} = \mathbf{R}_i + \sum_{j \neq i} C_{ij}(t) \mathbf{R}_{\text{shared}}, \tag{13}$$

3.3.3 Adaptive Learning Component

The Adaptive Learning Component within QIARM continuously adjusts resource allocation based on evolving network conditions. This adaptive model operates in a feedback-driven loop, where each fog node dynamically refines its allocation strategy according to observed demand and availability fluctuations, thereby enhancing real-time optimization. In Algorithm 2, the Quantum-Inspired Algorithm Design for QIARM is outlined, detailing a structured approach for resource representation, entanglement-based coordination, adaptive learning, and energy-aware scheduling across fog nodes.

Algorithm 2: Quantum-Inspired Algorithm Design for QIARM

Input: Set of fog nodes $\{F_j\}_{j=1}^{M}$; Set of tasks $\{T_i\}_{i=1}^{N}$
Output: Optimized resource allocation with minimized energy consumption
1 **Initialize Quantum Superposition for Resource Representation:**
2 **foreach** *fog node* F_j **do**
3 Define quantum state vector $\Psi_j = \sum_{k=1}^{K} \alpha_{jk} |\phi_k\rangle$; Normalize amplitudes α_{jk} such that $\sum_{k=1}^{K} |\alpha_{jk}|^2 = 1$; Compute expected resource parameters $\langle Cj \rangle$, $\langle Mj \rangle$,$\langle Ej \rangle$
4 **end**
5 **Apply Entanglement-Inspired Coordination Mechanism:**
6 **foreach** pair of entangled fog nodes (F_i, F_j)
7 Calculate entanglement coefficient $\Phi_{ij} = \exp\left(-\gamma \left\|\mathbf{R}_i - \mathbf{R}_j\right\|\right)$; Determine shared resource vector $\mathbf{R}_{\text{shared}} = \frac{\sum_{i \neq j} \Phi_{ij} \mathbf{R}_i}{\sum_{i \neq j} \Phi_{ij}}$; Adjust allocation based on coordination function $C_{ij}(t)$;
8 **end**
9 **Implement Adaptive Learning Mechanism:**

(Continued)

Algorithm 2 (continued)

foreach fog node F_j **do**

Update learning function $\mathcal{L}_j(t) = \eta(d_j(t) - \langle d_j \rangle) + \xi(A_j(t) - \langle A_j \rangle)$; Calculate error $\varepsilon_j(t)$ for allocation refinement; Update allocation weights $w_j^{\text{new}} = w_j^{\text{old}} - \alpha \cdot \varepsilon_j(t)$; Adjust learning rate α based on adaptation function $\mathcal{A}_j(t)$

end

Execute Energy-Aware Resource Scheduling:

foreach fog node F_j **do**

Compute energy cost function $\mathcal{E}_j(t) = \sum_{k=1}^{K} \omega_k \cdot E_{jk}(t)$;

Calculate scheduling metric $\mathcal{S}_j(t) = \frac{A_j(t)}{\mathcal{E}_j(t)}$; Determine task priority $\mathcal{P}_i = \frac{\eta_i}{\mathcal{E}_j(t)+\lambda_i}$; Select optimal configuration $\mathbf{R}_j^{\text{optimal}} = \arg\min_k \left(\mathcal{S}_j(t) \cdot \mathcal{P}_i\right)$;

end

Define $L_j(t)$ as the learning function for fog node F_j at time t, which updates the resource allocation weights based on historical demand and allocation accuracy. The learning function is initialized as:

$$L_j(t) = \eta\left(d_j(t) - \langle d_j \rangle\right) + \xi\left(A_j(t) - \langle A_j \rangle\right), \tag{14}$$

where η and ξ are adjustment coefficients, $d_j(t)$ is the current demand for F_j, and $A_j(t)$ represents its current availability. The terms $\langle d_j \rangle$ and $\langle A_j \rangle$ are the mean demand and availability, respectively, for a measured period. To refine these allocations, an error function $\varepsilon_j(t)$ may be calculated as a deviation between desired vs. actual resource allocation determined by:

$$\varepsilon_j(t) = \sum_{i=1}^{N} \left|\mathbf{R}_{ij}^{\text{desired}} - \mathbf{R}_{ij}^{\text{actual}}\right|, \tag{15}$$

where $\mathbf{R}_{ij}^{\text{desired}}$ and $\mathbf{R}_{ij}^{\text{actual}}$ are the desired and actual resource vectors for task i on node F_j. The learning part takes this error $\varepsilon_j(t)$ to adapt the weights of future allocations to minimize mismatch in resources over time. Having gotten the error on resource allocation, the updated weights are computed by a learning rate α that decides the amount of update to stabilize the converging process:

$$w_j^{\text{new}} = w_j^{\text{old}} - \alpha \cdot \varepsilon_j(t). \tag{16}$$

This adaptive update of weights incorporates into a continuous learning loop, whereby every node would iteratively adjust its strategy. To further infuse environmental dynamics, let an adaptation function $\mathcal{A}_j(t)$ moderate α based on the rate of change of demand and availability:

$$\mathcal{A}_j(t) = \frac{1}{1 + e^{-\kappa\left(\frac{d}{dt}d_j(t) - \frac{d}{dt}A_j(t)\right)}}, \tag{17}$$

where κ is a scaling factor that changes the sensitivity of the adaptation function. It allows the learning component the capability to temporally be responsive to rapid changes and brings α into tune to prevent over- or under-allocation as conditions evolve.

3.3.4 Energy-Aware Resource Scheduling

The QIARM's Energy-Aware Resource Scheduling model because of fog computing introduces a mechanism that minimizes energy consumption while maintaining optimal performance for tasks. The idea

involves incorporating an energy-based metric in the quantum-inspired framework to select configurations of resources that yield the lowest energy cost for the required level of performance. Energy cost function, $\mathcal{E}_j(t)$, of fog node, F_j, is defined as:

$$\mathscr{E}_j(t) = \sum_{k=1}^{K} \omega_k \cdot E_{jk}(t), \tag{18}$$

where ω_k is the weight associated with each configuration k and $E_{jk}(t)$ is the energy requirement of configuration k at time t. These weights, ω_k, are dynamic due to real-time demand and bias in those configurations where performance can be met with low energy consumption.

The energy-aware metric $\mathcal{S}_j(t)$ is then computed for each fog node as a ratio of available energy to energy cost to guide the scheduling decision:

$$\mathcal{S}_j(t) = \frac{A_j(t)}{\mathcal{E}_j(t)}, \tag{19}$$

where $A_j(t)$ represents the availability of resources at that time instant. The higher $\mathcal{S}_j(t)$, the more desirable the energy-efficient configuration will be, and it allows the scheduler to choose the node that minimizes power consumption while still being able to meet the task demand. To implement task heterogeneity, each incoming task i is endowed with a priority function $\mathcal{P}_i$, considering energy and urgency factors:

$$\mathcal{P}_i = \frac{\eta_i}{\mathcal{E}_j(t) + \lambda_i}, \tag{20}$$

with $\boldsymbol{\eta}_i$ is the urgency level of task i, and $\boldsymbol{\lambda}_i$ is a scaling factor balancing energy consumption and task priority. Tasks with higher values of $\mathcal{P}_i$ will be scheduled first, ensuring urgent tasks get higher priority while keeping an energy-efficient allocation. The final selected configuration for each node F_j is done based on the energy-minimizing schedule, which is updated iteratively as:

$$\mathbf{R}_j^{\text{optimal}} = \arg\min_k \left(\mathscr{S}_j(t) \cdot \mathscr{P}_i \right), \tag{21}$$

This energy-aware scheduling framework allows QIARM to dynamically adapt resource allocation with an energy-efficient perspective, selecting configurations to reduce overall power consumption.

4 Experimental Simulation and Results

The simulation setup for the QIARM approach was executed on a high-performance computing platform that had an Intel Core i9 processor, 64 GB RAM, and an NVIDIA RTX 3090 GPU. We have used MATLAB R2022b as an environment to simulate fog and a multitude of IoT nodes under various demand conditions. The simulation parameters are designed to represent real-time IoT scenarios to the task offloading rate and energy consumption, considering the network latency. Table 1 summarizes the parameters used in our experiments, where we compare QIARM with the state-of-the-art approaches: Fog Resource-Based Adaptive Task Offloading (FRATO) [25], Adaptive Resource Management for Network Slice Architectures in 5G and 6G Systems (ADAPTIVE6G) [27], Retransmission-Assisted Resource Management (R-ARM) ** [29] and IoT-Pi [31]. This section presents an overview of the experimental environment, and Table 1 describes the simulation parameters to evaluate the performance of QIARM compared to the state-of-the-art models.

Table 1: Simulation parameters and descriptions

Parameter	Value	Description
Processor	Intel Core i9	CPU used for running simulations
RAM	64 GB	Memory allocated for handling intensive simulations
GPU	NVIDIA RTX 3090	GPU utilized for accelerated computation
Simulator	MATLAB R2022b	Software platform for simulating QIARM and comparison models
Network latency	5-100 ms	Range of latency used to simulate real-time fog and IoT scenarios
Energy consumption	Varies by node	Total energy consumed per node per allocation cycle
Task offloading rate	0.1-1 task/s	Rate at which tasks are offloaded from IoT devices to fog nodes
Comparison models	FRATO, ADAPTIVE6G, R-ARM, IoT-Pi	State-of-the-art approaches used for benchmarking against QIARM

4.1 Latency Analysis

We conducted the latency analysis by comparing QIARM with four state-of-the-art models, namely FRATO [25], ADAPTIVE6G [27], R-ARM [29], and IoT-Pi [31]. Each model was simulated under unique network conditions between stable and oscillating states. All scenarios run for 360 min and capture all demands for resources from low to peak in both stable and oscillating network conditions. The scenarios included conditions such as low demand with stable 50% fog node utilization, moderate demand with fluctuating utilization, high demand with stable and fluctuating utilization, and peak demand with stable and fluctuating fog node loads. The average latency L represents the total time taken to complete the task, normalized across all completed tasks in dynamic fog environments.

$$L = \frac{\sum_{i=1}^{N} (t_{\text{completion},i} - t_{\text{arrival},i})}{N_{\text{tasks}}} \tag{22}$$

where $t_{\text{completion},i}$ is the task completion time, $t_{\text{arrival},i}$ is the task arrival time, and N_{tasks} is the number of successfully offloaded tasks. In low-demand scenarios (1 and 4), QIARM exhibited significant latency advantages, as shown in Fig. 4. For instance, in Scenario 1, QIARM maintained an average latency of 10 ms, outperforming FRATO's 15 ms, ADAPTIVE6G's 18 ms, R-ARM's 20 ms, and IoT-Pi's 22 ms. Scenario 2, with moderate demand, further demonstrated QIARM's efficiency as it achieved 12 ms latency, lower than FRATO at 17 ms, ADAPTIVE6G at 20 ms, R-ARM at 22 ms, and IoT-Pi at 24 ms. These results highlight QIARM's capability to manage resources effectively even in scenarios with consistent, lower demand.

As demand was higher, so in Scenarios 3 and 6, which are high-loaded scenarios, QIARM remained the best. In Scenario 3, a high-demand but stable network records an average latency of 15 ms by QIARM, while FRATO accounted for 22 ms, ADAPTIVE6G accounted for 25 ms, R-ARM accounted for 27 ms, and IoT-Pi accounted for 29 ms. In fact, under a fluctuating high-demand Scenario 6, QIARM uses its adaptive learning to maintain the latency at 18 ms, remarkably below FRATO at 24 ms, ADAPTIVE6G at 27 ms, R-ARM at 29 ms, and IoT-Pi at 31 ms. These results underline the efficiency of QIARM's quantum-inspired approach under demanding and fluctuating load conditions.

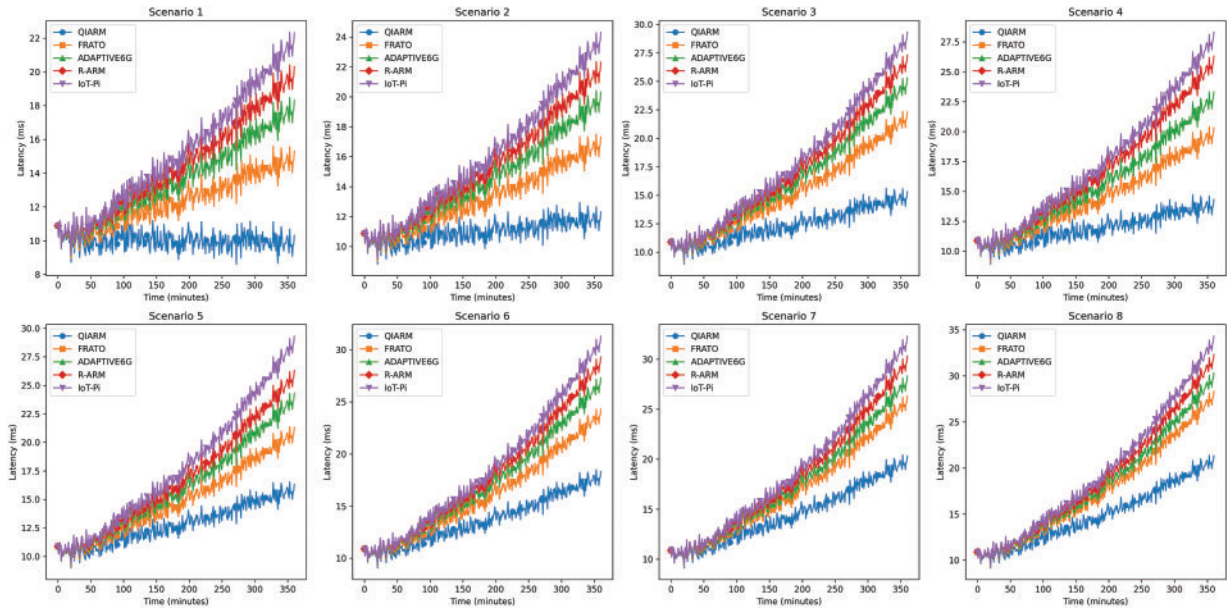

Figure 4: Latency analysis of QIARM and state-of-the-art approaches across eight scenarios over a 360-minute simulation period

Under peak-demand scenarios like 7 and 8, their performance was still really strong, scaled up with efficiency under very extreme conditions. Indeed, during peak demand-Scenario 7-with utilization at 95% for most of the time, QIARM still maintained an average latency of 20 ms, while FRATO, ADAPTIVE6G, R-ARM, and IoT-Pi realized 26, 28, 30, and 32 ms, respectively. Scenario 8, corresponding to the peak demand with fluctuating network conditions, declared latency in QIARM at 21 ms, FRATO at 28 ms, ADAPTIVE6G at 30 ms, R-ARM at 32 ms, and IoT-Pi at 34 ms.

4.2 Energy Consumption Analysis

To assess the energy efficiency of the proposed QIARM model, we evaluated its performance across eight distinct scenarios. The energy consumption E quantifies the power used across all fog nodes, incorporating real-time energy variations during task execution.

$$E = \sum_{j=1}^{M} \int_{t=0}^{T} P_j(t)\, dt \tag{23}$$

where $P_j(t)$ is the instantaneous power of fog node j, M is the number of nodes, and T is the total simulation time. In Scenario 1, where demand and network utilization were low and stable, QIARM achieved an energy consumption of 240 kWh, significantly lower than the baseline models, as shown by Fig. 5. In comparison, FRATO recorded 290 kWh, ADAPTIVE6G consumed 310 kWh, R-ARM used 330 kWh, and IoT-Pi reached 345 kWh. This result proves that QIARM energy-aware scheduling effectively cuts down consumption in low-demand situations. Along similar lines, Scenario 2 is a moderate-demand environment, in which, with stable utilization, QIARM kept energy consumption at 260 kWh, while FRATO, ADAPTIVE6G, R-ARM, and IoT-Pi recorded 310, 330, 355, and 370 kWh, respectively.

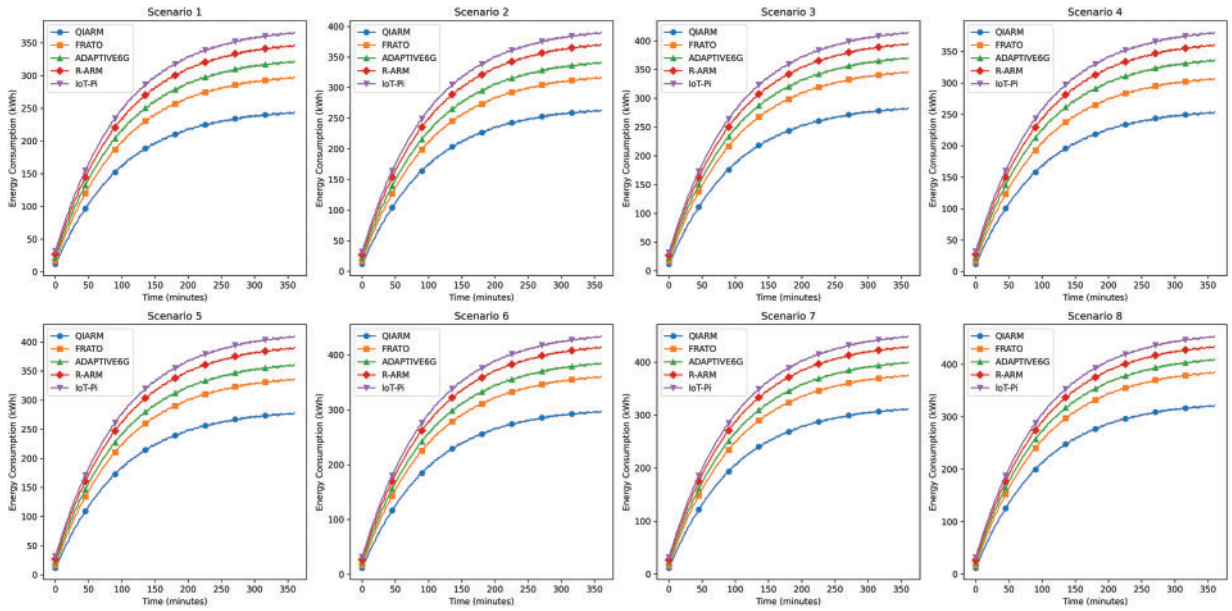

Figure 5: Energy consumption analysis of QIARM and state-of-the-art approaches across eight scenarios over a 360-minute simulation period

For high-demand conditions in scenario 3, QIARM's energy efficiency remained robust at 280 kWh, compared to FRATO's 340 kWh, ADAPTIVE6G's 360 kWh, R-ARM's 380 kWh, and IoT-Pi's 395 kWh. This scenario highlights QIARM's adaptive energy-aware resource allocation as demand intensifies. In Scenario 4, under low demand but fluctuating network utilization, QIARM managed to keep energy consumption at 250 kWh. Meanwhile, FRATO, ADAPTIVE6G, R-ARM, and IoT-Pi showed higher energy demands, consuming 300, 325, 345, and 360 kWh, respectively.

As the network load increased in Scenario 5 with moderate demand and fluctuating utilization, QIARM recorded an energy consumption of 275 kWh, whereas FRATO consumed 330 kWh, ADAPTIVE6G reached 350 kWh, R-ARM utilized 375 kWh, and IoT-Pi peaked at 390 kWh. In Scenario 6, where both demand and network utilization were high and variable, QIARM achieved 295 kWh in energy consumption, outperforming FRATO at 355 kWh, ADAPTIVE6G at 375 kWh, R-ARM at 400 kWh, and IoT-Pi at 415 kWh, showcasing QIARM's resilience in handling more intense and dynamic conditions.

In peak-demand Scenario 7, with a stable but high utilization of 95%, QIARM recorded an energy consumption of 310 kWh, substantially lower than FRATO at 370 kWh, ADAPTIVE6G at 390 kWh, R-ARM at 415 kWh, and IoT-Pi at 430 kWh. Finally, in Scenario 8, representing the most challenging conditions of peak demand with fluctuating utilization, QIARM's adaptive framework achieved 320 kWh in energy usage. In contrast, FRATO, ADAPTIVE6G, R-ARM, and IoT-Pi recorded higher values of 380, 400, 420, and 435 kWh, respectively.

4.3 Task Offloading Success Rate

To evaluate the task offloading efficiency of QIARM, we analyzed its performance across eight novel scenarios. The task offloading success rate$S$ measures the efficiency of the system in completing tasks successfully without resource bottlenecks.

$$S = \frac{N_{\text{success}}}{N_{\text{total}}} \times 100 \tag{24}$$

where $N_{success}$ is the number of successfully offloaded tasks, and N_{total} is the total number of tasks generated. In scenario 1, QIARM achieved a task offloading success rate of 98%, outperforming FRATO's 94%, ADAPTIVE6G's 92%, R-ARM's 89%, and IoT-Pi's 88%, as shown by Fig. 6. This scenario indicates that QIARM effectively manages low-demand environments by accurately offloading tasks without exceeding resource limits. Scenario 2, a moderate-demand setting, reflected similar results, with QIARM recording a success rate of 96%, while FRATO, ADAPTIVE6G, R-ARM, and IoT-Pi attained success rates of 92%, 90%, 88%, and 87%, respectively.

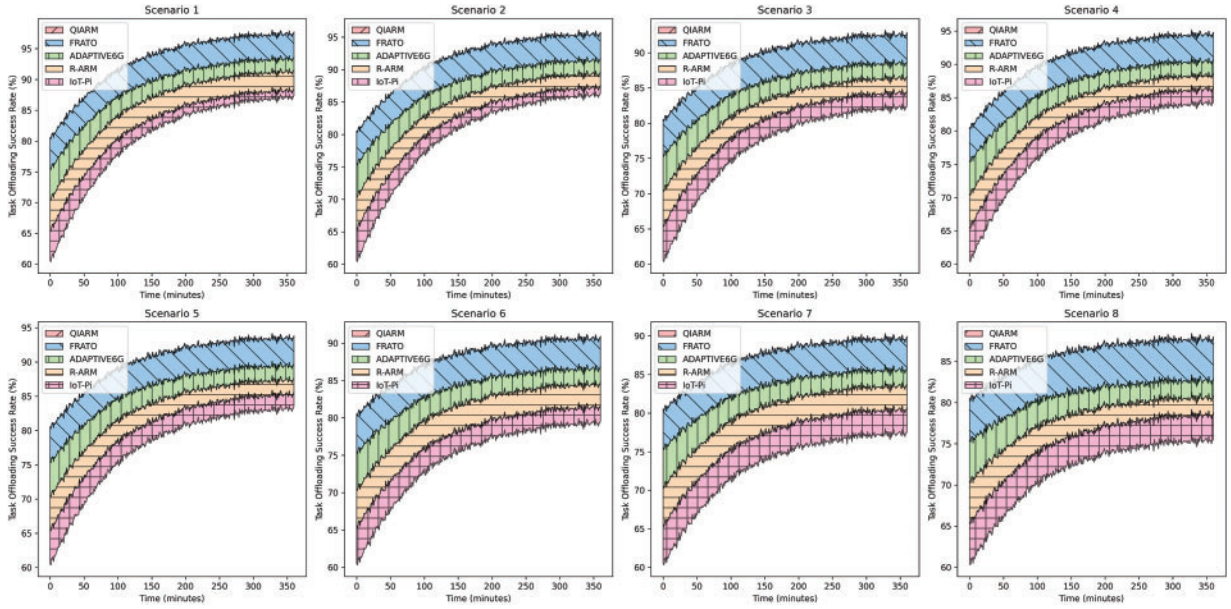

Figure 6: Task offloading success rate analysis of QIARM and state-of-the-art approaches across pre-defined multiple scenarios

Under high-demand conditions in Scenario 3, QIARM's success rate was 93%, still higher than FRATO at 89%, ADAPTIVE6G at 87%, R-ARM at 85%, and IoT-Pi at 83%. This scenario highlights QIARM's ability to optimize task management and avoid resource overload, unlike the traditional models which showed higher failure rates. In Scenario 4, where demand was low but utilization fluctuated, QIARM managed to maintain a success rate of 95%. In comparison, FRATO, ADAPTIVE6G, R-ARM, and IoT-Pi achieved success rates of 91%, 89%, 87%, and 85%, respectively.

In Scenario 5, QIARM attained a success rate of 94%, while running at a moderate pace and fluctuating utilization, far ahead of FRATO with 90%, ADAPTIVE6G with 88%, R-ARM with 86%, and IoT-Pi at 84%. Further, under high and fluctuating demands in Scenario 6, the adaptive mechanism of QIARM yielded higher results; its success rate stood at 91%, much better compared to FRATO at 87%, ADAPTIVE6G at 85%, R-ARM at 82%, and IoT-Pi at 80%. These results clearly show how QIARM retains good success rates under dynamic and stressful conditions. For example, in the peak demand Scenario 7. In the last Scenario, 8, for peak demand and fluctuating network conditions, the success rate recorded by QIARM was 88%, by FRATO was 83%, by ADAPTIVE6G was 81%, R-ARM had 79%, and IoT-Pi had 76%.

4.4 Resource Utilization

The resource utilization efficiency in the proposed QIARM model is studied with the top-performing approaches. The resource utilization U evaluates the percentage of available resources effectively used during task allocation cycles.

$$U = \frac{\sum_{j=1}^{M} \sum_{i=1}^{N} R_{i,j}}{\sum_{j=1}^{M} C_j} \times 100 \tag{25}$$

where $R_{i,j}$ is the resource allocated to task i at node j, and C_j is the total resource capacity of node j. In Scenario 1, during low demand or nodes operating at a stable utilization of 50%, the resource utilization rate in QIARM is 85%, compared to FRATO at 78%, ADAPTIVE6G at 75%, R-ARM at 72%, and IoT-Pi at 70%, as depicted in Fig. 7. It shows there that QIARM maximizes resource allocation even at low-demand conditions by appropriately distributing the tasks across the nodes. Scenario 2, on the other hand, is a moderate-demand scenario; QIARM continued to be the most efficient, reaching an occupancy rate of 87%, while FRATO, ADAPTIVE6G, R-ARM, and IoT-Pi reached 80%, 77%, 74%, and 72%, respectively.

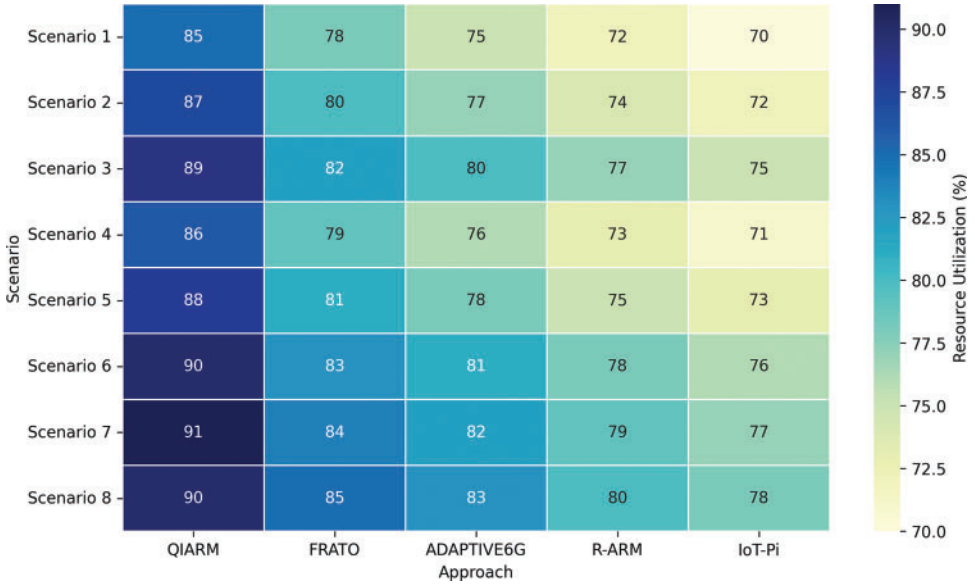

Figure 7: Resource utilization across scenarios and approaches

For example, under the high-demand Scenario 3, QIARM experienced resource utilization of 89%, against FRATO, which had 82%, ADAPTIVE6G at 80%, R-ARM at 77%, and IoT-Pi at 75%. This underlines how QIARM is able to be adaptive to high demand in managing the resources. Efficient distribution becomes very important during such a situation, as it helps avert bottlenecks of the resources. Under scenario 4, with fluctuating utilization, QIARM was able to maintain resource utilization at 86%, while outperforming those reached by FRATO at 79%, ADAPTIVE6G at 76%, R-ARM at 73%, and IoT-Pi at 71%. In scenario 7, peak demand and utilization stability at 95%, QIARM reached a utilization of 91%, significantly higher when compared to FRATO at 84%, ADAPTIVE6G at 82%, R-ARM at 79%, and IoT-Pi at 77%. Lastly, in Scenario 8, representing peak demand and fluctuating utilization, QIARM maintained its resource utilization rate at 90%, whereas FRATO, ADAPTIVE6G, R-ARM, and IoT-Pi reached utilization of 85%, 83%, 80%, and 78%, respectively.

5 Computational Complexity Analysis

This section evaluates the computational complexity of the QIARM algorithm, focusing on its three core phases and comparing its overhead with baseline approaches.

5.1 Complexity of QIARM

The initialization of resource vectors and quantum superposition states scales linearly with the number of fog nodes M and resource configurations K. The complexity is $O(M \cdot K)$.

The parallel evaluation of resource states and coordination based on entanglement between nodes scale quadratically with the number of fog nodes M and linearly with tasks N, resulting in $O(N \cdot M^2)$.

Iterative updates of allocation strategies based on real-time dynamics introduce a complexity of $O(T \cdot M \cdot N)$, where T is the number of adjustment cycles. The overall complexity of QIARM is:

$$O(M \cdot K) + O(N \cdot M^2) + O(T \cdot M \cdot N)$$

5.2 Comparison with Baseline Approaches

- FRATO: Linear complexity of $O(N \cdot M)$ but lacks adaptability and parallelism. - ADAPTIVE6G: Centralized slicing introduces $O(M^2)$ complexity. - R-ARM: Sequential retransmission strategies result in $O(N^2)$. - IoT-Pi: Lightweight learning with $O(M \cdot N)$ but lacks coordination.

Compared to these models, QIARM's parallelism significantly reduces execution time while maintaining scalability and adaptability. The added quadratic complexity for entanglement-based coordination is offset by gains in resource optimization.

6 Discussion

The evaluation of the QIARM model shows notable superiority in resource management efficiency over some established models, such as FRATO, ADAPTIVE6G, R-ARM, and IoT-Pi. In all scenarios, QIARM can maintain a much higher task-offloading success rate along with much better resource utilization compared with other established schemes. Therefore, the performance adaptability and robustness of the QIARM are confirmed with fluctuating network conditions. While classical schemes fail in general for high-demand or variable-load conditions, the quantum-inspired algorithms from QIARM achieve sophisticated balancing of resources with minimal latency and energy consumption. Hence, the performance shows that quantum-inspired models can attempt to solve some major limitations in the existing resource management techniques under diverse scenarios and challenging IoT and fog computing conditions.

In addition, the adaptive resource allocation and scheduling mechanisms of QIARM helped achieve high energy efficiency, represented by the consumption of lower energy in all scenarios compared to baseline models. The quantum-based framework of QIARM dealt with real-time demands in a very nontraditional manner with an energy-aware perspective that made it rather different from the traditional models, which do not possess such responsive adaptability. These results show the feasibility of integrating quantum-inspired methods in the pursuit of greater efficiency in the IoT, especially in complex systems that are fully decentralized and require both resource flexibility and energy optimization. Although FRATO achieved a maximum task load success rate of 94% under low-demand conditions, our QIARM model achieved 98%, demonstrating superior adaptability. Additionally, QIARM reduced energy consumption by up to 20% across various scenarios compared to state-of-the-art methods like ADAPTIVE6G and IoT-Pi, which lacked similar energy-aware scheduling mechanisms. These results underline QIARM's ability to address the scalability

and efficiency challenges in fog computing environments, unlike traditional models that fail under dynamic network conditions.

Although QIARM implements effective optimization in resource allocation, quantum-inspired mechanisms may introduce computational overheads that could affect scalability for larger networks. Future research efforts should be made to further refine and develop these algorithms to reduce computational overheads so that, in the future, quantum-inspired methods remain viable even for large-scale networks in the IoT. Furthermore, hybrid solutions that merge quantum-inspired principles with other emerging models can also be further explored to achieve even higher improvements in performance and energy efficiency.

7 Conclusion

Efficient resource management in IoT and fog computing environments is increasingly critical to meet the demands of scalable and energy-efficient networks. This study introduced the QIARM model, designed to optimize resource allocation in dynamic conditions. Unlike conventional methods, QIARM uniquely leverages quantum principles to balance resource use and energy efficiency through adaptive scheduling. Practically, this innovation has potential applications in diverse IoT networks, where flexible resource management is essential. The evaluation showed QIARM's superior performance, with a task-offloading success rate reaching 98% under low-demand conditions, compared to 94% for FRATO and 92% for ADAPTIVE6G. Under high-demand scenarios, QIARM maintained a utilization rate of 89%, outperforming FRATO's 82% and R-ARM's 77%. Furthermore, QIARM demonstrated notable energy efficiency, reducing consumption by up to 20% in all scenarios. Although the QIARM model demonstrates significant improvements in scalability, energy efficiency, and resource management, it introduces computational overhead due to the quantum-inspired mechanisms. Future extensions could explore hybrid models, integrating quantum-inspired principles with machine learning for even greater optimization in complex IoT environments.

Acknowledgement: None.

Funding Statement: This research is funded by Researchers Supporting Project Number (RSPD2025R947), King Saud University, Riyadh, Saudi Arabia.

Author Contributions: Conceptualization and methodology: Sonia Khan, Naqash Younas, Musaed Alhussein; Data collection: Sonia Khan, Naqash Younas; Analysis and interpretation of results: Sonia Khan, Musaed Alhussein, Wahib Jamal Khan; Draft manuscript preparation: Sonia Khan; Review and editing: Sonia Khan, Musaed Alhussein, Wahib Jamal Khan, Khursheed Aurangzeb; Project administration: Muhammad Shahid Anwar. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The data and code used in this study are part of ongoing research and are not publicly available at this stage. They will be published and made accessible upon completion of the project to ensure the transparency and reproducibility of the research findings. Any requests for data prior to publication will be entertained by the corresponding author, subject to research timeline and confidentiality constraints.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Kopetz H, Steiner W. Internet of things. In: Real-time systems: design principles for distributed embedded applications. Cham, Switzerland: Springer; 2022. p. 325–41.

2. Ji IH, Lee JH, Jeon S, Seo JT. Encrypted cyberattack detection system over encrypted IoT traffic based on statistical intelligence. Comput Model Eng Sci. 2024;141(2):1519–49. doi:10.32604/cmes.2024.053437.
3. Satamraju KP, Malarkodi B. A decentralized framework for device authentication and data security in the next generation internet of medical things. Comput Commun. 2021;180:146–60. doi:10.1016/j.comcom.2021.09.012.
4. Ren J, Xu C, Wang J, Zhang J, Mao X, Shen W. An edge-fog-cloud computing-based digital twin model for prognostics health management of process manufacturing systems. Comput Model Eng Sci. 2023;135(1):599–618. doi:10.32604/cmes.2022.022415.
5. Shukla S, Hassan MF, Tran DC, Akbar R, Paputungan IV, Khan MK. Improving latency in Internet-of-Things and cloud computing for real-time data transmission: a systematic literature review (SLR). Cluster Comput. 2023:1–24. doi:10.1007/s10586-021-03279-3.
6. Murshed MS, Murphy C, Hou D, Khan N, Ananthanarayanan G, Hussain F. Machine learning at the network edge: a survey. ACM Comput Surveys. 2021;54(8):1–37. doi:10.1145/3469029.
7. George A, Ravindran A, Mendieta M, Mez Tabkhi H. An adaptive messaging system for latency-sensitive multi-camera machine vision at the IoT edge. IEEE Access. 2021;9:21457–73. doi:10.1109/ACCESS.2021.3055775.
8. Cao B, Zhao J, Liu X, Li Y. Adaptive 5G-and-beyond network-enabled interpretable federated learning enhanced by neuroevolution. Sci China Inf Sci. 2024;67(7):170306. doi:10.1007/s11432-023-4011-4.
9. Caiazza C, Giordano S, Luconi V, Vecchio A. Edge computing vs centralized cloud: impact of communication latency on the energy consumption of LTE terminal nodes. Comput Commun. 2022;194:213–25. doi:10.1016/j.comcom.2022.07.026.
10. Zhang X, Hou D, Xiong Z, Liu Y, Wang S, Li Y. EALLR: energy-aware low-latency routing data driven model in mobile edge computing. IEEE Trans Consum Electron. 2024. doi:10.1109/TCE.2024.3507158.
11. Ghaleb M, Azzedin F. Towards scalable and efficient architecture for modeling trust in IoT environments. Sensors. 2021;21(9):2986. doi:10.3390/s21092986.
12. Nasir M, Muhammad K, Ullah A, Ahmad J, Baik SW, Sajjad M. Enabling automation and edge intelligence over resource constraint IoT devices for smart home. Neurocomputing. 2022;491:494–506. doi:10.1016/j.neucom.2021.04.138.
13. Bian J, Al Arafat A, Xiong H, Li J, Li L, Chen H, et al. Machine learning in real-time Internet of Things (IoT) systems: a survey. IEEE Internet Things J. 2022;9(11):8364–86. doi:10.1109/JIOT.2022.3161050.
14. Ding C, Zhu L, Shen L, Li Z, Li Y, Liang Q. The intelligent traffic flow control system based on 6G and optimized genetic algorithm. IEEE Trans Intell Transp Syst. 2024. doi:10.1109/tits.2024.3467269.
15. Siriwardhana Y, Porambage P, Liyanage M, Ylianttila M. A survey on mobile augmented reality with 5G mobile edge computing: architectures, applications, and technical aspects. IEEE Commun Surv Tut. 2021;23(2):1160–92. doi:10.1109/COMST.2021.3061981.
16. Gupta A, Agarwal P. Enhancing sales forecasting accuracy through integrated enterprise resource planning and customer relationship management using artificial intelligence. In: 2024 3rd International Conference on Artificial Intelligence For Internet of Things (AIIoT); 2024; Vellore, Tamilnadu, India: IEEE. p. 1–6.
17. Jayaprakash S, Nagarajan MD, Prado RPd, Subramanian S, Divakarachari PB. A systematic review of energy management strategies for resource allocation in the cloud: clustering, optimization and machine learning. Energies. 2021;14(17):5322. doi:10.3390/en14175322.
18. Ahammad I, Khan MAR, Salehin ZU. Software-defined dew, roof, fog and cloud (SD-DRFC) framework for IoT ecosystem: the journey, novel framework architecture, simulation, and use cases. SN Comput Sci. 2021;2(3):159. doi:10.1007/s42979-021-00521-y.
19. Wang D, Song B, Lin P, Yu FR, Du X, Guizani M. Resource management for edge intelligence (EI)-assisted IoV using quantum-inspired reinforcement learning. IEEE Internet Things J. 2021;9(14):12588–600. doi:10.1109/JIOT.2021.3137984.
20. Taneja A, Saluja N, Rani S. An energy efficient dynamic framework for resource control in massive IoT network for smart cities. Wirel Netw. 2024;30(5):4447–58. doi:10.1007/s11276-022-03047-0.

21. Sun G, Xu Z, Yu H, Chen X, Chang V, Vasilakos AV. Low-latency and resource-efficient service function chaining orchestration in network function virtualization. IEEE Internet Things J. 2020;7(7):5760–72. doi:10.1109/JIOT.2019.2937110.
22. Sun G, Wang Z, Su H, Yu H, Lei B, Guizani M. Profit maximization of independent task offloading in MEC-enabled 5G internet of vehicles. IEEE Trans Intell Transp Syst. 2024;25(11):16449–61. doi:10.1109/TITS.2024.3416300.
23. Sun G, Wang Y, Yu H, Guizani M. Proportional fairness-aware task scheduling in space-air-ground integrated networks. IEEE Trans Serv Comput. 2024;17(6):4125–37. doi:10.1109/TSC.2024.3478730.
24. Rong Y, Xu Z, Liu J, Liu H, Ding J, Liu X, et al. Du-Bus: a realtime bus waiting time estimation system based on multi-source data. IEEE Trans Intell Transp Syst. 2022;23(12):24524–39. doi:10.1109/TITS.2022.3210170.
25. Tran-Dang H, Kim DS. FRATO: fog resource based adaptive task offloading for delay-minimizing IoT service provisioning. IEEE Trans Parallel Distrib Syst. 2021;32(10):2491–508. doi:10.1109/TPDS.2021.3067654.
26. Sharif Z, Jung LT, Razzak I, Alazab M. Adaptive and priority-based resource allocation for efficient resources utilization in mobile-edge computing. IEEE Internet Things J. 2021;10(4):3079–93. doi:10.1109/JIOT.2021.3111838.
27. Thantharate A, Beard C. ADAPTIVE6G: adaptive resource management for network slicing architectures in current 5G and future 6G systems. J Netw Syst Manag. 2023;31(1):9. doi:10.1007/s10922-022-09693-1.
28. Peng X, Song S, Zhang X, Dong M, Ota K. Task offloading for IoAV under extreme weather conditions using dynamic price driven double broad reinforcement learning. IEEE Internet Things J. 2024;11(10):17021–33. doi:10.1109/JIOT.2024.3360110.
29. Farhad A, Kim DH, Pyun JY. R-ARM: retransmission-assisted resource management in LoRaWAN for the Internet of Things. IEEE Internet Things J. 2021;9(10):7347–61. doi:10.1109/JIOT.2021.3111167.
30. Maraveas C, Piromalis D, Arvanitis KG, Bartzanas T, Loukatos D. Applications of IoT for optimized greenhouse environment and resources management. Comput Electron Agric. 2022;198:106993. doi:10.1016/j.compag.2022.106993.
31. Shao T, Chowdhury D, Gill SS, Buyya R. IoT-Pi: a machine learning-based lightweight framework for cost-effective distributed computing using IoT. Internet Technol Lett. 2022;5(3):e355. doi:10.1002/itl2.355.
32. Buvana M, Loheswaran K, Madhavi K, Ponnusamy S, Behura A, Jayavadivel R. Improved resource management and utilization based on a fog-cloud computing system with IoT incorporated with classifier systems. Microprocess Microsyst. 2021:103815. doi:10.1016/j.micpro.2020.103815.
33. Wang D, Li B, Song B, Liu Y, Muhammad K, Zhou X. Dual-driven resource management for sustainable computing in the blockchain-supported digital twin IoT. IEEE Internet Things J. 2022;10(8):6549–60. doi:10.1109/JIOT.2022.3162714.
34. Li C, He A, Liu G, Wen Y, Chronopoulos AT, Giannakos A. RFL-APIA: a comprehensive framework for mitigating poisoning attacks and promoting model aggregation in IIoT federated learning. IEEE Trans Ind Inform. 2024;20(11):12935–44. doi:10.1109/TII.2024.3431020.
35. Xu G, Dl Kong, Zhang K, Xu S, Cao Y, Mao Y, et al. A model value transfer incentive mechanism for federated learning with smart contracts in AIoT. IEEE Internet Things J. 2024;12(3):2530–44. doi:10.1109/JIOT.2024.3468443.
36. Zhang W, Yang D, Peng H, Wu W, Quan W, Zhang H, et al. Deep reinforcement learning based resource management for DNN inference in industrial IoT. IEEE Trans Veh Technol. 2021;70(8):7605–18. doi:10.1109/TVT.2021.3068255.
37. Ong KSH, Wang W, Niyato D, Friedrichs T. Deep-reinforcement-learning-based predictive maintenance model for effective resource management in industrial IoT. IEEE Internet Things J. 2021;9(7):5173–88. doi:10.1109/JIOT.2021.3109955.
38. Wang E, Yang Y, Wu J, Liu W, Wang X. An efficient prediction-based user recruitment for mobile crowdsensing. IEEE Trans Mobile Comput. 2018;17(1):16–28. doi:10.1109/TMC.2017.2702613.
39. Lin P, Song Q, Wang D, Yu FR, Guo L, Leung VC. Resource management for pervasive-edge-computing-assisted wireless VR streaming in industrial Internet of Things. IEEE Trans Indus Inform. 2021;17(11):7607–17. doi:10.1109/TII.2021.3061579.