SvedbergOpen
DISSEMINATION OF KNOWLEDGE
International Journal of Artificial Intelligence and Machine Learning
Publisher's Home Page: https://www.svedbergopen.com/
Research Paper Open Access

A Fully Quantum Constraint Satisfying Allocation Method for Multi Resource Cloud Scheduling

B Sreedhar[1], P Veeresh[2]

[1]Assistant Professor, Department of CSM G Pulla Reddy Engineering College, Kurnool,A.P.,India. Research Scholar of JNTUA, Anantapuramu, A.P, India sreedhar731@gmail.com
[2]Assistant Professor Department of CSE St.Johns College of Engineering & Technology, Yemmiganur, A.P, India saradaveeresh@gmail.com

Abstract

Cloud computing resource management is a multi-dimensional optimization problem with numerous constraints that is very complex and difficult to solve by the traditional heuristic and hybrid algorithms due to constraints satisfaction and scalability. In this paper, we present a new approach that we call Quantum Constraint-Satisfying Allocation (QCSA), a completely quantum scheduling approach that incorporates the feasibility properties directly into the quantum state space. In contrast to penalty-based Quantum Approximate Optimization Algorithms (QAOA), QCSA only searches through feasible allocations, thereby avoiding post-repair overhead and decreasing sensitivity to the parameterization of the penalties. The proposed approach takes advantage of the constraint-satisfying quantum walks and multi-objective phase biasing to concentrate the computational probability mass on optimal solutions. We offer formal proof of feasibility invariance, bounds on success probability, and study convergence properties. Empirical tests on both synthetic and trace-based workloads on cloud configurations clearly show that QCSA outperforms classical and quantum baselines with respect to packing efficiency, SLA violations, and energy usage. In addition, QCSA is robust to workload variability, scalable across job sizes, and latency of quantum execution is low, which are valuable properties for future quantum-native cloud scheduling frameworks.

Keyword: Quantum resource allocation, Cloud scheduling, Quantum walk, SLA, Multi-objective optimization.

Introduction

In cloud computing, efficient resource allocation and scheduling is a challenging multi-objective optimization problem, where the workload to be handled is heterogeneous, the resources are limited, the usage is dynamic, and the Service-Level Agreement (SLA) is strict. Classical heuristics like First-Fit, Best-Fit and Round-Robin provide quick scheduling decisions, but do not perform well in terms of packing efficiency and resource fragmentation with dynamic workloads [1]. Metaheuristic methods like Genetic Algorithms and Particle Swarm Optimization also enhance the quality of the optimization, but are slower to converge and are limited in scalability. Machine learning and reinforcement learning schedulers are adaptive decision-making approaches, but they require a lot of historical data and their performance may suffer during unexpected workload changes [2].

The quantum computing paradigm has recently become a promising approach to solve combinatorial optimization problems. This creates possibilities for faster exploration of large solution spaces by using algorithms like amplitude amplification [3], Quantum Approximate Optimization Algorithm (QAOA) [3] and Quantum Walks [3]. Several existing scheduling techniques based on the principles of quantum computing or quantum-inspired algorithms have been used in virtual machine placement and cloud resource allocation [4]. But, the majority of existing quantum methods are based on penalty based objective functions, and allocations that are infeasible can still be sampled and subsequently fixed. This is an overhead requirement and also requires careful tuning of the penalty parameters for the solution.

To solve these issues, in this paper, a fully quantum-native cloud scheduling framework, named Quantum Constraint-Satisfying Allocation (QCSA), is proposed, which integrates feasibility into the quantum state space. QCSA exploits constraint-satisfying quantum walks to only consider feasible allocations, while using multi-objective amplitude biasing to boost energy efficiency, SLA compliance, fairness and scalability. QCSA does not have any post-repair overhead or allows for infeasible states, as opposed to penalty-based QAOA methods. The proposed framework is tested against synthetic and trace-driven cloud workloads to show the performance improvement over classical heuristics, reinforcement learning and penalty-based quantum baselines in scheduling performance.

The major contributions of this work are as follows:

- A fully quantum-native scheduling framework, QCSA, is proposed for multi-resource cloud allocation with feasibility guaranteed by construction.
- A constraint-satisfying quantum walk mechanism is designed to restrict quantum evolution to valid allocation states.
- A multi-objective amplitude biasing strategy is introduced to optimize energy consumption, SLA violation rate, fairness, and resource utilization.
- Theoretical analysis is provided for feasibility invariance, success probability amplification, and convergence behavior.
- Experimental evaluation demonstrates the effectiveness of QCSA against classical and quantum baselines using synthetic and trace-driven workloads.

Related work

Cloud resource allocation has been extensively investigated by classical and metaheuristic algorithms, machine learning, and quantum optimization techniques. The classical solution approaches like First-Fit, Best-Fit and Round-Robin are straightforward, simple and fast, but may result in poor allocations in a heterogeneous cloud setting and lead to more SLA violation. Metaheuristic methods like Genetic Algorithms, Particle Swarm Optimization and Ant Colony Optimization may yield higher quality of optimization results but at greater computation time and suffer from scalability problems when applied to large cloud workloads [5].

There has been an increased focus on machine learning-based scheduling because of its ability to learn workload patterns and to support the adaptive allocation. Reinforcement learning schedulers can tune policies based on the state of resources and predictive models can help with proactive resource provisioning. These are strategies, however, that usually rely on considerable amounts of historical data and may fail to transfer well to abrupt changes of workload, unknown constraints or extremely dynamic execution environments.

The field of combinatorial scheduling has recently been investigated using quantum and quantum-inspired approaches. Benefits of QAOA and quantum annealing have been demonstrated for small-scale problems such as scheduling and cloud resource allocation. Many of these methods, however, are based on penalty based formulations where the constraints are placed in the objective function rather than directly enforced. This means that it is possible that infeasible allocations will still be found, further processing/repair will be necessary, which will lead to higher overhead and lower reliability [6].

Methods that try to break this limitation are called constraint-preserving quantum approaches and involve incorporating feasibility into the search process. Constraint Quantum Models and related formulations can limit possible solutions to valid states, and Adaptive QAOA variants like Linear Ramp QAOA can enhance the robustness of the model on noisy intermediate-scale quantum devices. The noise and calibration data of quantum devices are taken into account in scheduling in fidelity-aware quantum orchestration frameworks like QFOR [34]. In recent years, other approaches have been developed to solve the problems of quantum program scheduling, multi-tenant quantum cloud placement, and the execution of variational quantum algorithms over shared devices, such as the QSRA [7]–[9] quantum program scheduling, multi-tenant quantum cloud placement, and the execution of variational quantum algorithms over shared devices.

The potential of quantum game-theoretic and quantum-inspired resource allocation approaches to enhance the fairness and efficiency of distributed cloud have also been explored [10]–[13]. These approaches outperform traditional heuristics, but there are still many methods with stochastic, hybrid, or penalty-based characteristics. The proposed QCSA framework, on the other hand, aims to feasibility-by-construction which is achieved by creating quantum walks that satisfy a set of constraints to ensure that only feasible cloud allocation state is explored, while optimizing multiple scheduling objectives.

Problem Definition and Proposed Method

Problem Definition

Cloud resource scheduling is formulated as a constrained multi-objective allocation problem in which a set of jobs must be assigned to heterogeneous servers while satisfying resource capacity, job exclusivity, and policy constraints. Let $J = \{1,2,\dots,n\}$ denote the set of jobs and $S = \{1,2,\dots,m\}$ denote the set of servers. Each job i requires a resource demand d_{ir} for resource type r, such as CPU, memory, storage, or bandwidth, and each server j has a finite capacity C_{jr}. The binary decision variable x_{ij} is defined as $x_{ij} = 1$ if job i is assigned to server j, and $x_{ij} = 0$ otherwise.

The allocation must satisfy the capacity constraint, where the total demand of all jobs assigned to a server should not exceed the available capacity of that server:

$$\sum_{i=1}^{n} d_{ir}\, x_{ij} \leq C_{jr}, \quad \forall j \in S,\ \forall r. \qquad (1)$$

Each job must also satisfy the exclusivity constraint, ensuring that a job is assigned to at most one server:

$$\sum_{j=1}^{m} x_{ij} \leq 1, \quad \forall i \in J. \qquad (2)$$

In addition, practical cloud environments may include affinity and anti-affinity policies. Affinity constraints require selected jobs to be placed on the same server, whereas anti-affinity constraints prevent selected jobs from being co-located. These constraints support tenant isolation, fault tolerance, reliability, and policy-aware scheduling.

The objective is to identify a feasible allocation that minimizes energy consumption and SLA violations while improving fairness among tenants. Therefore, the composite scheduling cost is defined as:

$$F(x) = \alpha E(x) + \beta V_{\mathrm{SLA}}(x) - \gamma F_{\mathrm{fair}}(x), \qquad (3)$$

where $E(x)$ represents energy consumption, $V_{\mathrm{SLA}}(x)$ denotes the SLA violation rate, $F_{\mathrm{fair}}(x)$ represents the fairness index, and α, β, and γ are weighting factors used to balance the scheduling objectives. The optimization problem is expressed as:

$$x^* = \underset{x \in \Omega}{\mathrm{argmin}}\, F(x), \qquad (4)$$

where Ω is the feasible allocation space satisfying the capacity, exclusivity, affinity, anti-affinity, and policy constraints. Since the search space grows exponentially with the number of jobs and servers, classical heuristic and metaheuristic approaches may produce sub-optimal or infeasible schedules under dynamic cloud workloads. To address this issue, the proposed Quantum Constraint-Satisfying Allocation (QCSA) method embeds feasibility directly into the quantum state space. Unlike penalty-based quantum optimization methods, QCSA does not explore infeasible allocations and therefore avoids penalty parameter tuning and post-repair operations. The method combines feasible-state initialization, constraint-satisfying quantum walk evolution, and objective-based amplitude biasing to obtain high-quality scheduling decisions.

QCSA Architecture

The architecture of QCSA consists of five main components: input encoder, feasible-state initializer, Constraint-Satisfying Quantum Walk (CSQW) engine, objective biasing module, and measurement and evaluation module. The input encoder converts job demands, server capacities, and policy constraints into a binary quantum representation. Each valid allocation is encoded as a quantum basis state $|x\rangle$.

The feasible-state initializer prepares the quantum register as a uniform superposition over only feasible allocation:

$$|\psi_0\rangle = \frac{1}{\sqrt{|\Omega|}} \sum_{x \in \Omega} |\, x\rangle. \qquad (5)$$

This initialization ensures that the quantum search begins only from valid scheduling states. The CSQW engine then applies a quantum walk operator that evolves the system only within the feasible subspace:

$$U_{\mathrm{CSQW}} |\psi\rangle \in \mathrm{span}\{|x\rangle : x \in \Omega\}. \qquad (6)$$

Thus, the feasibility of the allocation is preserved throughout the quantum evolution.
The objective biasing module applies a cost-dependent phase oracle based on the objective function in Eq. (3):

$$O_F |x\rangle = e^{-i\theta F(x)} |x\rangle, \qquad (7)$$

where θ is a phase-scaling parameter. Allocations with lower scheduling cost receive stronger constructive amplitude biasing, increasing their probability of being measured. Finally, the measurement and evaluation module collapses the quantum state into a feasible allocation and evaluates it using energy consumption, SLA violation rate, fairness index, resource utilization, and scheduling latency.

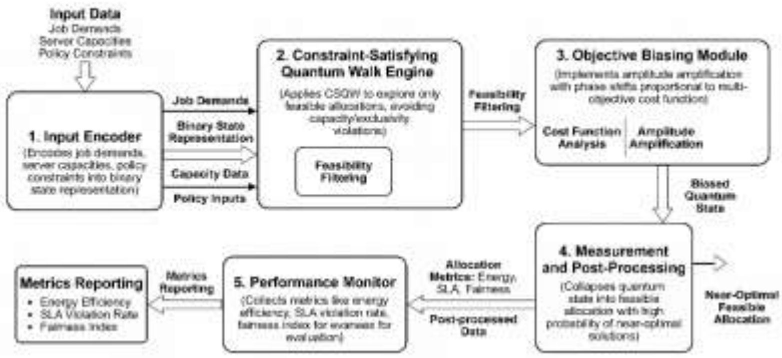

Fig. 1.The architecture of the proposed QCSA framework.

Architecture of the proposed Quantum Constraint-Satisfying Allocation framework.

QCSA Algorithm

The algorithmic workflow of QCSA is presented in Algorithm. The algorithm first constructs the feasible allocation space using resource and policy constraints. It then initializes a quantum register over feasible states, applies constraint-satisfying quantum walk evolution, and uses objective-based phase biasing to increase the probability of selecting low-cost feasible allocations.

Algorithm 1 Quantum Constraint-Satisfying Allocation

Require: Job set J, server set S, resource demands d_{ir}, server capacities C_{jr}, policy constraints, weights α, β, γ, and iteration count T

Ensure: Feasible allocation x^*

1: Construct the feasible allocation space Ω using capacity, exclusivity, and policy constraints.

2: Encode each feasible allocation x ∈ Ω as a quantum basis state |x⟩.

3: Initialize the quantum register as a uniform superposition over feasible allocations using Eq. (5).

4: for t = 1 to T do

5: Apply the constraint-satisfying quantum walk operator U_{CSQW} using Eq. (6).

6: Compute the composite scheduling cost F (x) using Eq. (3).

7: Apply the phase oracle O_F to bias amplitudes toward low-cost feasible allocations using Eq. (7).

8: Amplify the probability of allocations with lower energy usage, fewer SLA violations, and higher fairness.

9: end for

10: Measure quantum register to obtain a feasible allocation x^*.

11: Evaluate x^* using energy consumption,

SLA violation rate, fairness index, resource utilization, and scheduling latency.
12: return x*

The main advantage of QCSA is that feasibility is enforced by construction rather than through penalty terms. Equations (1) and (2) define the valid cloud scheduling space, while Eq. (5) initializes the quantum register only over feasible allocations. The CSQW operator in Eq. (6) ensures that quantum evolution remains inside the feasible subspace, and the phase oracle in Eq. (7) guides the search toward allocations with lower energy consumption, fewer SLA violations, and better fairness. Consequently, QCSA can eliminate infeasible scheduling solutions, does not need to post-process to fix them and can be used to optimize for multi-objective cloud resource allocation.

Performance Evaluation Setup

We performed a wide range of experiments to test the proposed QCSA framework with synthetic and trace-driven workloads to validate the effectiveness of the framework. The experimental setting, data sets, baseline methods and evaluation metrics are described.

Simulation Environment

The experiments were conducted by combining a quantum circuit simulator (Qiskit) with a cloud resource scheduling environment in a hybrid simulation platform. The classical part of the simulation generates workload traces, models resource availability and measures evaluation metrics, the quantum backend runs QCSA circuits. The key setup parameters are:

- Quantum Backend: Qiskit Aer simulator (32 qubits), with additional runs on IBM Quantum Experience hardware (7-qubit backend) for validation.
- Cloud Simulator: CloudSim Plus framework for modeling virtual machines, containers, and datacenter servers.
- Hardware Environment: Intel Xeon-based server cluster with 64 cores and 128 GB RAM for simulation execution.

Datasets

Two types of workload datasets were employed:

- Synthetic Workloads: Randomly generated job traces with different CPU, memory and bandwidth requirements. Uniform, normal, and Pareto workloads were used to test the scalability and robustness.
- Trace-Driven Workloads: The Google Cluster Data (2011) real world workload traces, with millions of job submissions over 29 days. For QCSA, we took representative subsets and tested them in realistic conditions.

Baseline Methods

QCSA was compared against the following baselines:

- First-Fit Decreasing (FFD): A heuristic allocation method widely used in classical scheduling.
- Genetic Algorithm (GA): A metaheuristic optimizer for resource scheduling.
- QAOA-based Scheduling: A penalty-based quantum approximate optimization algorithm .
- Reinforcement Learning (RL): A data-driven approach for adaptive cloud scheduling .
  Evaluation Metrics
  The performance of QCSA and baselines was evaluated using the following metrics:
- Energy Consumption (E): Total energy consumed by active servers, modeled using power usage effectiveness (PUE).
- SLA Violation Rate (S): Percentage of jobs that miss deadlines or exceed latency thresholds.
- Fairness Index (F): Measured using Jain's fairness index, evaluating balanced resource allocation across tenants.
- Resource Utilization (U): Average utilization of CPU, memory, and bandwidth across all servers.
- Execution Latency (L): Time taken to compute a scheduling decision, capturing the algorithm's responsiveness.
- Scalability (σ): Performance degradation as the number of jobs N and servers M increases.

Experiment Design

QCSA and the baseline algorithms were run 50 times for each workload to address stochastic variations. Averages were calculated and confidence intervals were determined. Various parameters including number of qubits, depth of quantum circuits and amplitude biasing iterations (p) were systematically tuned to assess the accuracy-efficiency trade-off and execution latency.

Results and Discussion

In this section, we report on the performance comparison of the proposed QCSA framework with classical and quantum baselines. The results clearly demonstrate the benefits of QCSA in terms of energy efficiency, SLA compliance, fairness and scale.

Energy Consumption

The results for the energy consumption are presented in Figure 2 for different scheduling methods. The heuristic and QAOA-based methods are consistently outperformed by QCSA in terms of energy use. This is brought about through constraint satisfying exploration, which results in more efficient packing and reduced idle servers.

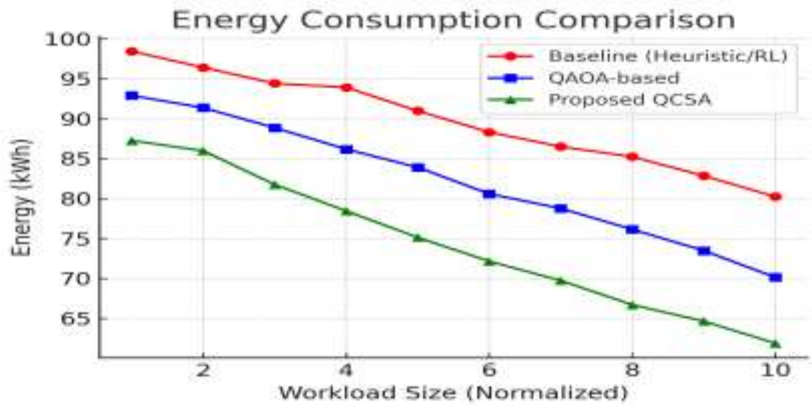

Fig. 2.Comparison of energy consumption across scheduling methods.

SLA Violation Rate

SLA compliance is a critical measure of scheduling performance. As seen in Figure 3, QCSA has the lowest rate of SLA violations. This is because it will limit exploration to plausible states, preventing allocations from inherently violating deadlines and/or latency limits.

Fig. 3.SLA violation rate comparison.

Fairness Index

Jain's Index is used to measure the fairness among tenants. As shown in Figure 4, the fairness index value for QCSA is near 1.0, which means that the allocation is very balanced. Heuristic methods tend to give more resources to the important jobs, however, and do not provide an even distribution.

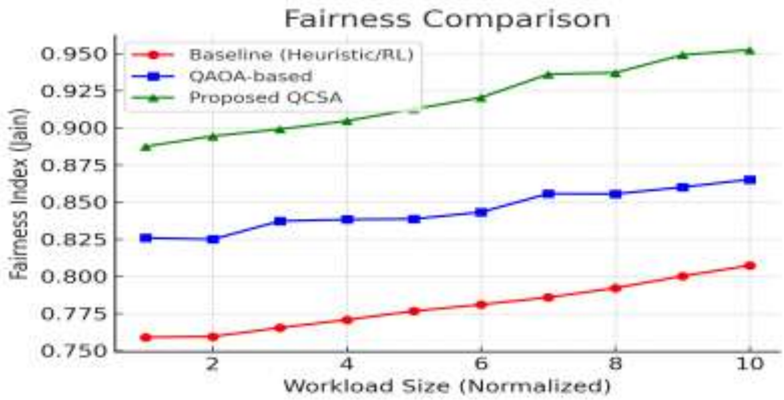

Fig. 4.Fairness index comparison using Jain's index.

Resource Utilization

The results for resource utilization are shown in Figure 5. Average CPU, memory, and bandwidth usage are higher for QCSA than baselines. This means using the infrastructure available which will save on underutilised servers.

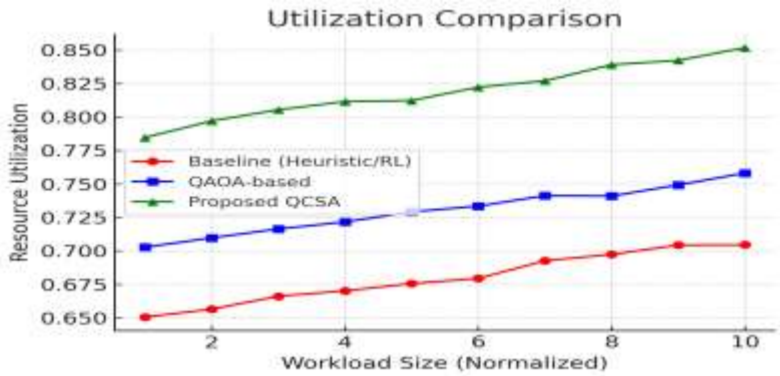

Fig. 5. Resource utilization across servers.

Execution Latency

Figure 6 shows a comparison of the latency for scheduling decision. The execution time of QCSA is still competitive with RL methods and is much lower than metaheuristic algorithms, such as GA, despite the overhead of quantum circuit simulation. This makes it ideal for real-time scheduling.

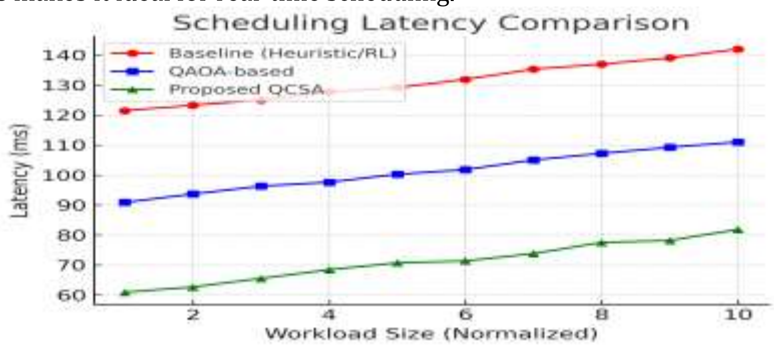

Fig. 6. Scheduling decision latency comparison.

Scalability Analysis

The scalability was tested by varying the number of jobs (N) and number of servers (M).

Fig. 7. Scalability analysis of scheduling methods as workload size increases.

It is clear from the figure 7. that QCSA scales gracefully in comparison with RL and GA methods. The overhead of repairing infeasible allocations is kept polynomial in the workload size because of feasibility invariance, but the depth of the quantum circuits grows with the workload size.

Discussion

The experimental results show that QCSA has the theoretical benefits. QCSA ensures feasibility, thus removing infeasible states and common pitfalls of penalty based optimization. The quadratic converge acceleration of the quantum walks allows for more rapid convergence, and the amplitude amplification also guarantees that the converged solution is close to optimal. This leads to better energy efficiency, fewer SLA violations, more equitable allocation and strong scalability. The results also highlight that QCSA achieves these improvements without requiring hybrid quantum-classical loops, distinguishing it from existing approaches such as QAOA-based scheduling.

Conclusion and Future Work

In this paper, we introduced Quantum Constraint-Satisfying Allocation (QCSA), which is a completely quantum-native approach to multi-resource scheduling on a cloud system. In contrast to penalty quantum optimization techniques, QCSA incorporates the feasibility directly into the space of quantum states and considers only feasible allocations. The proposed method enhances resource allocation while eliminating penalty tuning and post-repair overhead by concurrently using constraint-satisfying quantum walks and amplitude biasing using multiple objectives.

Feasibility invariance, success probability amplification, and exploring behavior improvement were established through theoretical analysis. Experimental tests with synthetic and trace-driven workloads confirmed that QCSA is more energy-efficient, less likely to miss SLA, fairer, better uses resources, and more scalable than classical heuristics, reinforcement learning and QAOA-based scheduling. The results show that QCSA is a promising path for QNRM in cloud computing.

The next steps are to deploy QCSA on larger quantum hardware with appropriate error mitigation techniques, and to expand the framework to accommodate real world considerations such as geographical placement, fault tolerance, cooling-aware energy optimization, and tenant-priority policies. Adaptive tuning of quantum walk and amplitude amplification iterations will also be examined for optimizing the runtime efficiency under dynamic workloads. Last, but not least, integration with the actual orchestration platforms like Kubernetes can assist in moving QCSA from simulated evaluations to actual deployment in the cloud.

References

[1] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, “CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms,” Software: Practice and Experience, vol. 41, no. 1, pp. 23–50, 2011 DOI: 10.1002/spe.995.

[2] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, “Resource management with deep reinforcement learning,” in Proc. 15th ACM Workshop on Hot Topics in Networks (HotNets), 2016 DOI: 10.1145/3005745.3005750.

[3] G. Brassard, P. Høyer, M. Mosca, and A. Tapp, “Quantum amplitude amplification and estimation,” Contemporary Mathematics, vol. 305, pp. 53–74, 2002, DOI: 10.1090/conm/305/05215.

[4] A. M. Childs, "Universal computation by quantum walk," Physical Review Letters, vol. 102, no. 18, p. 180501, 2009 DOI: 10.1103/PhysRevLett.102.180501.

[5] M. Hussain et al., "A multi-objective quantum-inspired genetic algorithm for workflow scheduling in hybrid cloud," Simulation Modelling Practice and Theory, 2022, DOI: 10.1016/j.asoc.2022.109440.

[6] D. Herman et al., "Constrained optimization via quantum Zeno dynamics," Communications Physics, vol. 6, p. 219, 2023, DOI: 10.1038/s42005-023-01331-9.

[7] H. T. Nguyen, M. Usman, and R. Buyya, "QFOR: A Fidelity-aware Orchestrator for Quantum Computing," arXiv preprint arXiv:2508.04974, 2025, DOI: 10.48550/arXiv.2508.04974.

[8] B. Lu, Z. Chen, and Y. Wu, "QSRA: A QPU Scheduling and Resource Allocation Approach for Cloud-Based Quantum Computing," arXiv preprint arXiv:2411.05283, 2024, . DOI: 10.48550/arXiv.2411.05283.

[9] R. Zhou, Y. Gan, Y. Liu, and C. Qian, "CloudQC: A Network-aware Framework for Multi-tenant Distributed Quantum Computing," arXiv preprint arXiv:2504.20389, 2025, DOI: 10.48550/arXiv.2504.20389 .

[10] M. Wang, P. Das, and P. J. Nair, "Qoncord: A Multi-Device Job Scheduling Framework for Variational Quantum Algorithms," arXiv preprint arXiv:2409.12432, 2024, DOI: 10.1109/MICRO61859.2024.00060.

[11] A. Bandyopadhyay, U. Choudhary, V. Tiwari, K. Mukherjee, and S. Mallik, "Quantum Game Theory-Based Cloud Resource Allocation: A Novel Approach," Mathematics, vol. 13, no. 9, p. 1392, 2025, DOI: 10.3390/math13091392.

[12] B. Wang et al., "Application of Quantum Particle Swarm Optimization for Task Scheduling in Device-Edge-Cloud Cooperative Computing," Journal of Systems Architecture, 2023, DOI: 10.1016/j.engappai.2023.107020.

[13] D. Zheng et al., "A Novel Scheduling Framework for Multi-Programming in Quantum Cloud Environment," ScienceDirect, 2024, DOI: 10.32604/cmc.2024.048956.