Received: January 29, 2024. Revised: March 1, 2024.

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Towards Quantum Computing-Inspired Evolutionary Algorithm for Optimized Cloud Resource Management

Visalaxi G[1]* **Muthukumaravel A[2]**

[1]*Department of Computer Science and Engineering,*
Bharath institute of higher Education and Research, Chennai, India
[2]*Department of Arts and Science, Bharath Institute of Higher Education and Research, Chennai, India*
* Corresponding author's Email: visalaxikumar@gmail.com

Abstract: Resource allocation plays a pivotal role in Cloud Computing (CC), significantly impacting its overall performance. In order to fairly spread workload among servers, network ports, and hard drives, task scheduling is a critical module of CC. Cloud computing experiences request overloading as a result of dynamic computing over the internet. To address this challenge, a novel Cloud based Load balancing using Quantum artificial bee colony Optimization Algorithm has been proposed for task scheduling that can significantly enhance the effectiveness of cloud computing operations. The technique focuses on two important aspects of cloud computing: resource allocation and task scheduling. By achieving load balancing between servers, the proposed CLUQOA (Cloud based Load balancing Using Quantum artificial bee colony Optimization Algorithm) improves reliability and lowers expenses, latency, and response times. To show how effective it is, key performance parameters including makespan, resource usage, task migration, task execution time, and response time are assessed. The results of comparative comparisons with other approaches, such as HUNTER (Holistic resoUrce maNagemenT technique for Energy-efficient cloud computing using aRtificial intelligence), FPNSO (Flower Pollination based Non-dominated Sorting Optimization), and ProHPA (Proactive Hybrid Pod Autoscaling), demonstrate the superiority of CLUQOA in terms of improving resource utilization (35.6%, 26.4%, and 13.9%, respectively) and reducing makespan (11.02%, 9.6%, and 10.4%, respectively).

Keywords: Quantum computing, Task scheduling, Artificial bee colony algorithm, Quantum cloud computing, Qubits.

1. Introduction

Cloud computing stands as a prominent approach for delivering applications on demand via the internet. It has emerged as a transformative paradigm for delivering on-demand applications over the Internet, revolutionizing the way computational resources are provisioned and utilized [1, 2]. Even still CC has a lot of potential advantages, effective source administration and allocation in cloud schemes are still being researched and discussed. Infrastructure as a Service (IaaS) in particular is essential, as virtual machines dynamically meet the various demands of cloud users [3,4]. Efficient resource allocation—including CPU, memory, and storage—is critical to cloud computing success in order to maintain user satisfaction and optimal performance [5].

Task scheduling is unity of the key components of CC. Task scheduling techniques need to be optimized if cloud computing is to function as efficiently as possible [6]. Cloud computing uses the Internet to aggregate and manage massive amounts of idle computing power [7]. Its main objective is to give consumers services quickly and consistently. Users only need to send tasks to the cloud data focus on terminals to use cloud platforms to meet a range of needs and ultimately obtain processing results. Optimum mission development and cloud source administration are key topics in cloud computing research, as they enhance bandwidth usage and job

execution efficiency [8]. Figure 1 depicts the framework for cloud-based data analysis.

Efficient distribution and management of computational workloads across available resources is a key challenge in cloud computing. Load balancing is a crucial method that assures efficient resource use, decreases response times, and avoids individual components within a cloud infrastructure from being overburdened [9]. In cloud systems, load balancing is essential to achieve high performance and dependability. Load balancing helps to avoid bottlenecks and guarantees that no single resource is overworked by effectively dividing incoming requests or tasks among several servers or virtual machines [10].

On the other hand, a cloud computing system is extremely complex because it consists of hundreds of cloud resource nodes. Cloud computing is still experiencing a lot of teething pains. Because resources and activities are dynamically heterogeneous, job scheduling and equitable lab or distribution remain major challenges in cloud computing systems [11,12]. It is challenging to ensure the continued operation of every computing node due to frequent issues with hardware malfunctions, linkage failures, and system overload [13]. A novel CLUQOA has been presented to address these problems. The primary contributions of this paper are as follows:

- This paper lies in introducing a novel Cloud based Load balancing using Quantum artificial bee colony Optimization Algorithm named CLUQOA, for optimized cloud resource management.
- The proposed method addresses the trials of task scheduling and source distribution in cloud computing. CLUQOA leverages Quantum-Artificial Bee Colony Optimization, integrating quantum principles to enhance algorithm diversity and efficiency.
- The proposed CLUQOA technique balances the load across servers, increases reliability, and effectively lowers costs, latency, and response times.
- The makespan, resource consumption, task migration, task execution time, and response time are all have been used to examine its performance.

The following sections of this work are organized as follows. A synopsis of the relevant work is given in Section II of the article. The suggested methodology is provided in Section III. Section IV provides a detailed presentation of the experiment data that show how effective the recommended strategy is. Section V serves as the article's final conclusion.

2. Literature survey

The supply of computing and IT services, as well as the control and enhancement of resource and data consumption, have all benefited greatly from CC. Numerous professionals have devised a number of strategies to keep load balancing between virtual servers. We have covered a couple of these algorithms in this article.

In 2022, Tuli, S., et al., [14] introduced HUNTER, a comprehensive reserve organization strategy for justifiable CC that is built on artificial intelligence (AI). HUNTER outperforms state-of-the-art

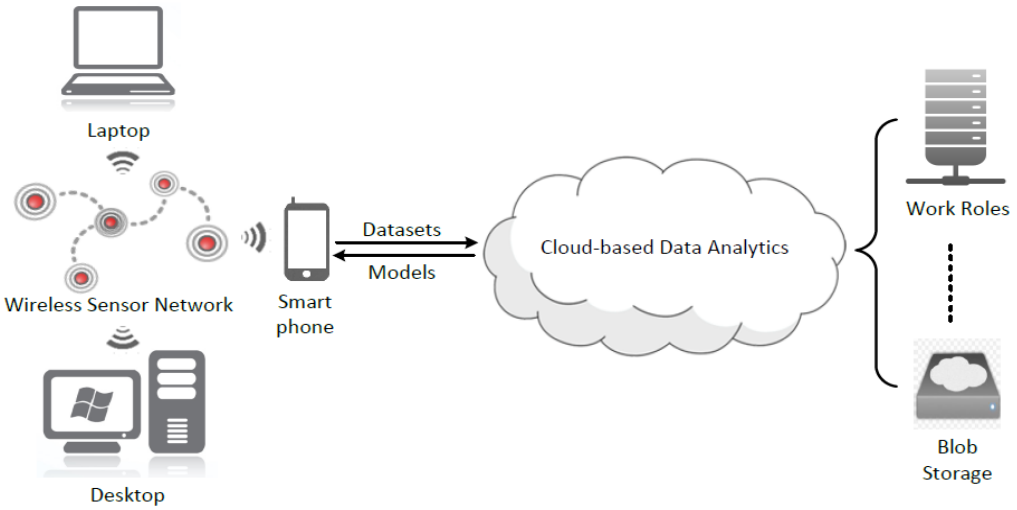


Figure. 1 Cloud-based data analysis framework

baselines by up to 12 percent in terms of energy usage, SLA violation, scheduling time, cost, and temperature. It might be difficult for the suggested HUNTER approach to handle a variety of intricate real-world situations.

In 2022, Saxena, D., et al., [15] suggested a framework for fault-tolerant elastic resource management (FT-ERM) that induces maximum accessibility in servers and virtual machines (VMs) to solve issues with load imbalance, performance degradation, and excessive power consumption. Energy consumption and VM migration are decreased by 62.4%, and 88.6% correspondingly, Implementing reactive fault tolerance and security measures like N-Version Programming and VM trust adds complexity and resource overhead to the framework.

In 2023, Singh, A.K., et al., [16] suggested Flower Pollination based Non-dominated Sorting Optimization (FPNSO), which lowers energy consumption and maximizes resource use, hence lowering carbon emissions in the data center. The outcomes demonstrate a significant improvement in resource utilization of 35.79% along with notable reductions in power, execution time, and carbon with improvements of up to 17.03%, 12.42%, and 33.21% respectively.The disadvantages are low security, low reliability and low trust which potentially increase complexity.

In 2023, Jeong, B., et al., [17] suggested Proactive Hybrid Pod Autoscaling (ProHPA), which lowers resource overallocation and reacts quickly to erratic workloads. In comparison to conventional HPA, ProHPA's performance was assessed, and when initial resources were overallocated, memory average utilization and CPU increased by 42.52%, and 23.39% correspondingly. Furthermore, in the case of inadequate resource allocation, ProHPA did not display excess in contrast to conservative HPA.

In 2023, Shuaib, M., et al., [18] presented a comprehensive approach to discourse all source distribution issues in the Internet of Things, dubbed dynamic energy-efficient load balancing (DEELB). Many tests are carried out, including an analysis of bandwidth usage. While other currently in use methods such as DEBTS, ELBS, DEERA, and EEFO utilized 1200.15 kbps, 1300.65 kbps, 1500.82 kbps, and 1700.91 kbps of bandwidth, respectively. The drawbacks are increased complexity and resource overhead, potentially hindering its scalability.

In 2023, Senthil Kumar, A.M., et al., [19] suggested HGCSBAT, a unique job allocation algorithm that combines the BAT algorithm and Cat Swarm Optimization. The planned HGCSBAT algorithm's outcomes are assessed and associated to the well-known CSO and BAT algorithms. In terms of availability and throughput, HGCSBAT performs better than BAT algorithms, Cat Swarm Optimization, and Genetic. The paper doesn't address limitations or challenges of the suggested Algorithm.

In 2023, Surya, K. and Rajam, V.M.A., [20] suggested two models—the Hybrid Cascade of Regression and Markov model for Resource Contention Prediction (CRMRCP) and the Dynamic Markov model for Reserve Argument Prediction in Edge Cloud (DMRCP)—that forecast resource contention at the edge servers. Results show DMRCP had 52.9% fewer VM migrations than first-order Markov and 21.1% less than second-order. The drawback are increased complexity and resource overhead in predicting memory and storage requirements for computation across edge servers.

Some of the disadvantages of the previously discussed approaches for balancing the loads in virtual machines include their inability to do computational offloading, their inability to lower costs when deadlines are relaxed, their dearth of substantial amounts of useful data, etc. To overcome these challenges a novel CLUQOA technique has been proposed which is detailly explained in the following section.

3. Cloud based load balancing using quantum artificial bee colony optimization algorithm

In this paper, a novel Cloud based Load balancing using Quantum artificial bee colony Optimization Algorithm (CLUQOA) has been proposed for task scheduling that can significantly enhance the effectiveness of cloud computing operations. The artificial bee algorithm and quantum computing are projected to concurrently resolve all of the scheduling task's problems, including reaction time, waiting time, and turnaround time. Conventional schedulers have been developed with such criteria; however, starvation issues are rarely a concern. When dealing with a big cloudlet, the proposed methodology combined with the optimum quantum technique appears to shorten the waiting time. The proposed method focuses on optimizing the mechanism of identifying the state of resources as they have specific and adaptive outcomes. The large-scale, diverse deployment of resources, on the other hand, brings major challenges to cloud computing's complicated resource management strategies. Figure 2. Displays the overview of the projected CLUQOA technique.

3.1 Quantum-ABC optimization

In the proposed CLUQOA, a quantum depiction of nutriment sources is used to both reduce the special complexity and improve algorithm diversity. Nonetheless, the addition of quantum interference should ensure effective search space exploitation. An effective substitute for computation with strong computation capabilities is quantum computing. quantum algorithms that make use of the fundamental ideas and principles of quantum computing, such as

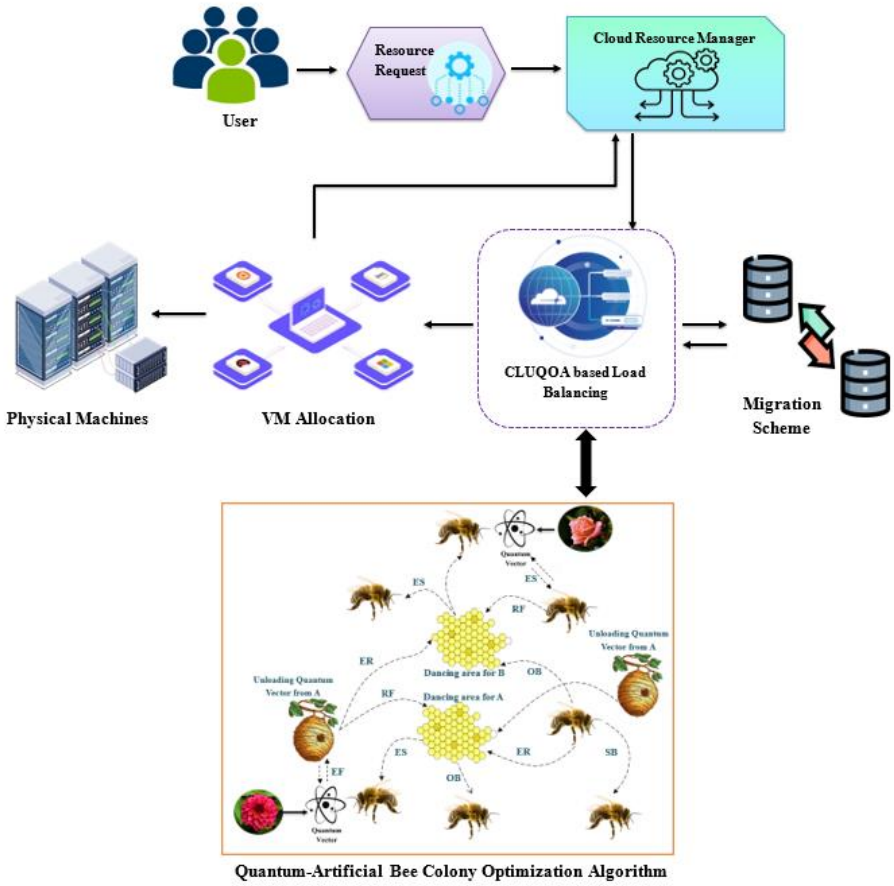


Figure. 2 Overview of the proposed CLUQOA technique

Table 1. Lists the notations

Notation	Definition
R_{ji}	final solution
S_j	The solution that has to be perturbed
$\emptyset_{ji}$	Randomly Generated Number
$X(\gamma_X, \delta_X)$	Qubit to be tampered
$Y(\gamma_Y, \delta_Y)$	Resulting qubit
f_j	The excellence charge of the proposed resolution
$bst\ (j)$	Optimal Result
m	Entire quantity of food sources
ft_j	Solution j's fitness
t_j	Time at which the task has been completed.
Res_{ut}	Resource utilization
$L_{Ti}(L_y)$	Total time taken for execution of y^{th} task.
C_{Tik}	computational time for task y,

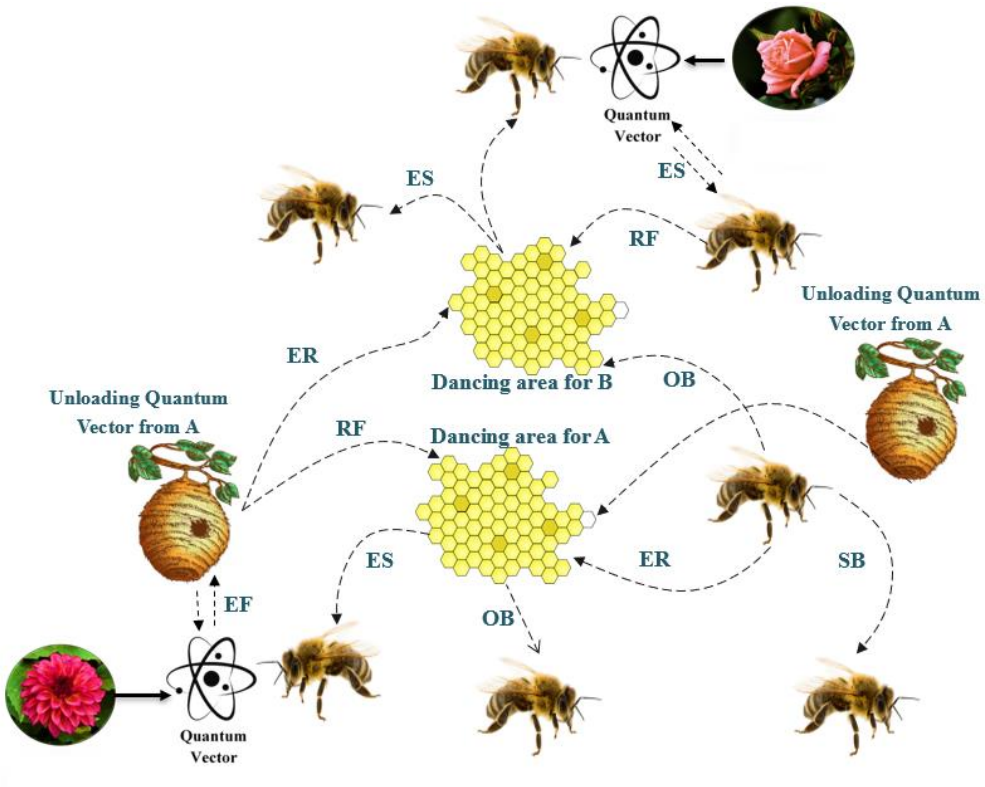


Figure. 3 Quantum-Artificial Bee Colony Optimization Algorithm

quantum gates, quantum bits, quantum registers, and measurement operations. The basic informational unit of quantum computing is the quantum bit, or qubit for short. A tiny device known as a qubit symbolizes the superposition of the states 0 and 1. A collection of qubits, or an arbitrary arrangement of n qubits, is called a quantum register. Figure 3. Represents the Quantum-Artificial Bee Colony Optimization Algorithm.Table 1 shows lists the notations used in this paper along with their Definition.

A quantum representation is used to ensure that the population of solutions has greater diversity. In this arrangement, a quantum vector represents each food source (solution). The suggested technique replicates the resulting steps over 3 phases, which are corresponding to the ABC phases, until a halt form is encountered. First, the inhabitants of entities are initialized with N quantum trajectories that are initialized arbitrarily. Next, the ailment of union of the sum of chances at apiece qubit is verified. The steps are Dimension, Assessment, Greedy Assortment, Quantum ABC operator and Quantum interfering. The phasesare Working bee phase, Spectator Bee phase, and Emissary phase: Regeneration. These stages are detailed in the succeeding.

3.1.1. Working bee phase

Dimension: By projecting a quantum solution's qubits one at a time onto a binary space, this operative permits a quantum binary result to be converted into a classical one. Assessment: Afterward changing the binary result acquired from gauging the quantum resolution into an actual worth, the Assessment worker uses the suitability algorithm in Eq. (1) to determine the binary solution's fitness.

$$ft_j = \begin{cases} \dfrac{1}{1+f_j}, & f_j \geq 0 \\ 1 + sab(f_j), & f_j < 0 \end{cases} \quad (1)$$

The excellence charge of the proposed resolution to the given optimization issue is represented by the symbol fi in the equation. Greedy Assortment: A avaricious assortment among the best solution discovered thus far and the achieved one, based on their fitness values, occurs when the suitability of the key in query is determined. The more fit answer gets retained. Quantum ABC operator: This operator is essential to searching the hunt space. It makes it possible to find novel quantum solutions. Actually, it's the quantum equivalent of the ABC operator that's mentioned in Eq. (2).

$$R_{ji}\begin{pmatrix}\gamma_{ji}\\ \delta_{ji}\end{pmatrix} = S_{ji}\begin{pmatrix}\gamma_{ji}\\ \delta_{ji}\end{pmatrix} + \emptyset_{ji}\left(S_{ji}\begin{pmatrix}\gamma_{ji}\\ \delta_{ji}\end{pmatrix} - S_{jh}\begin{pmatrix}\gamma_{ji}\\ \delta_{ji}\end{pmatrix}\right) \quad (2)$$

If R_{ji} is the final solution, S_{jh} is an additional randomly selected solution from the population, and S_j is the solution that has to be perturbed. A random index from {1,2,...,D} is denoted by j, where D is the dimension of the issue. Within the range [-1, 1], the coefficient $\emptyset_{ji}$ is a randomly generated number. Quantum interfering: Each qubit in a given solution can be altered in the route of the consistent bit worth of the finest result initiate so far through a process called "quantum interference". Here's how this operator operates:

Let $X(\gamma_X, \delta_X)$represent the qubit to be tampered with, finest the finest worth the swarm has found thus far, and a numeral constant. Then, using Eqs. (3) and (4), determine the resulting qubit $Y(\gamma_Y, \delta_Y)$.

$$\gamma_X = \frac{bst\ (j) + k \times \gamma_X}{k+1} \quad (3)$$

$$\delta_Y = \sqrt{1-\gamma_Y^2} \quad (4)$$

The values of γ and δ will grow and decrease respectively if the charge $bst\ (j)$ of the optimal result, to which the delayed qubit X is steered, equals 1. A larger chance of receiving a 1 is therefore obtained. In a similar way, when $bst\ (j)$ = 0, γ will automatically decrease and δ will automatically increase. Quantum interference plays a fundamental role in quantum-inspired algorithms, and the suggested CLUQOA is no different. The optimal food source that the colony's bees have all discovered is used by this operator to help lead the bees. It allows the colony's bees to take advantage of their common knowledge.

3.1.2. Spectator bee phase

The spectator bee phase uses the same procedures as were used in the watching bee phase. The primary distinction among the two phases is that in the first, all of the solutions are rationalized, whilst in the second, only the selected solutions need to be updated. The basic ABC algorithm's probability, as stated in Eq. (5), is used to determine which solutions should be updated at this stage.

$$Py_j = \frac{ft_j}{\sum_m ft_m} \quad (5)$$

If m is the entire quantity of food sources and ft_j is the solution j's fitness.

3.1.3. Emissary phase: regeneration

A regeneration of certain quantum people is used at this point. The sustenance sources (quantum results) that have not altered for a predefined number of phases—referred to as the emissary phase or bound—are swapped by new food sources that are generated at random. Eq. (6) states that the qubits in these solutions must adhere to the requirement.

$$|\gamma|^2 + |\delta|^2 = 1 \quad (6)$$

Each bee decreases to a single state and receives a value of 0 or 1 with a probability greater than one upon seeing a quantum state. With the mapping matrix, each row denoting a single element with a value of one, tasks would be assigned to a single data centre at a time. The proposed method collects the tasks and data centre ID for the input data. The mapping between the data centers and the tasks is the output that makes it easier for tasks to get to the selected data centers.

4. Results and discussion

In this section, the popular simulation application CloudSim is used to evaluate the efficacy of the modern tactics chosen for this study. An Intel Core i5-8500 quad-core workstation with 8 GB of RAM and 3.0 GHz system speed was employed to carry out the simulation tests. Google cluster dataset is used to verify the proposed CLUQOA technique efficiency. Before going into a discussion of the results, assessment metrics, and baseline procedures for the experiment are explained. The efficacy of the proposed model is compared to HUNTER [14], FPNSO [16], and ProHPA [17], techniques respectively.

4.1 Dataset description

The Google Cluster dataset comprises data collected from a large-scale cluster of Google servers. It includes information such as CPU usage, memory usage, disk I/O, and network traffic. This dataset is valuable for analyzing resource utilization patterns, optimizing performance, and developing efficient scheduling algorithms for data center management.

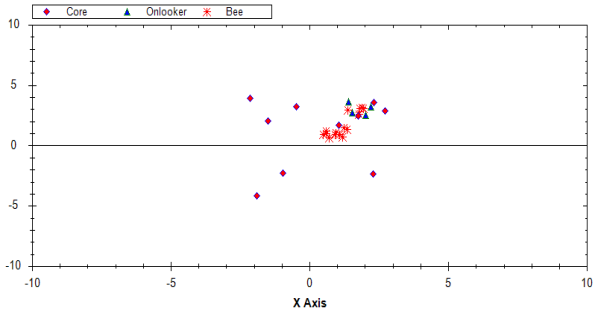


Figure. 4 Implementation of ABC

4.2 Implementation result of QABC

A simulation using an artificial bee colony algorithm was devised, where a graph composed of interconnected nodes represented the environment.

This artificial bee was designed to interpret and generate its own pheromone on these arcs, illustrated in Figure 4. Each bee's decision-making process involved probabilities, determining the likelihood of bee "b" finding the next neighbouring node "j" based on the stored information of its previous node "i."

4.3 Performance metrics

In this part, the main performance metrics that are looked at in the experimental evaluation are covered. There is discussion of definitions and their respective importance. Relevant metrics to assess in this domain include makespan, resource consumption, task migration, task execution time, and response time.

4.3.1. Performance based on makespan

The quantity of period wanted to finish all jobs is called a makespan. Eq. (6) contains the formula for calculating the makespan.

$$MS = MAy\left(T_b(t_j)\right) \quad (7)$$

Where, MS represent the makespan and t_j is the time at which the task has been completed.

A Makespan comparison of our proposed CLUQOA technique with current HUNTER [14], FPNSO [16], and ProHPA [17], techniques are shown in Figure 5. Our method shows a significant reduction in Makespan across different job numbers, consistently outperforming competitors. The graph illustrates how our approach to task scheduling ensures optimal resource use due to its higher efficiency and scalability. The proposed method achieves better makespan of 11.02%, 9.6% and 10.4% than HUNTER [14], FPNSO [16], and ProHPA [17] techniques respectively.

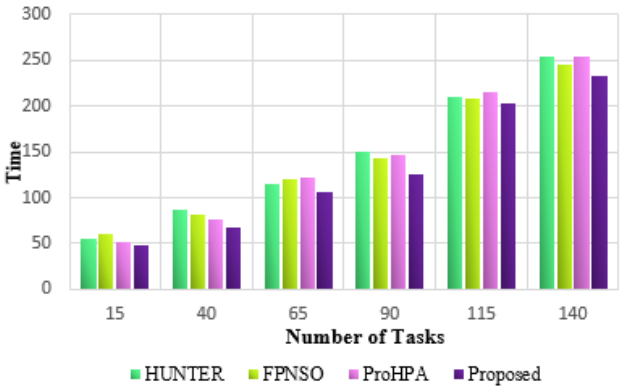


Figure. 5 Comparision in terms of Makespan

4.3.2. Performance based on resource utilization

By optimizing resource consumption, cloud computing may avoid needless loading that squanders the resources of individual hosts. It is also feasible to compute the average resource use with the equation given in (7).

$$Res_{ut} = [(ORes - ERes) \times (ORes)] \times 100 \quad (8)$$

Here, Res_{ut} represents the resource utilization, $ORes$ represents the original resource and $ERes$ represents the Existing resource.

Figure 6 displays a comprehensive comparison of resource utilization between our proposed CLUQOA technique with current HUNTER [14], FPNSO [16], and ProHPA [17] techniques. Our solution excels across diverse task numbers, showcasing superior resource efficiency. The graph underscores the effectiveness of proposed CLUQOA in optimizing resource utilization, ensuring optimal performance across varying task complexities. The proposed method achieves higher resource utilization of 35.6%, 26.4% and 13.9% than HUNTER [14], FPNSO [16], and ProHPA [17] techniques respectively.

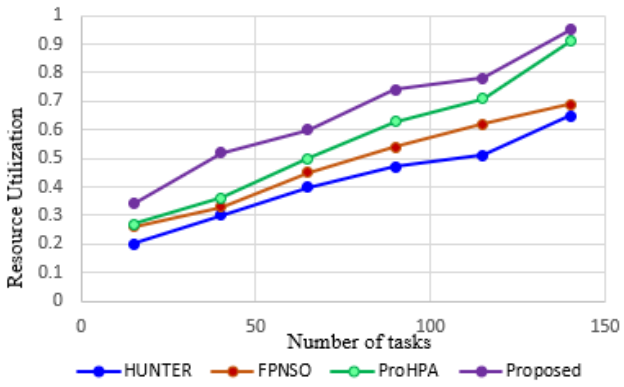


Figure. 6 Comparision in terms of resource utilization

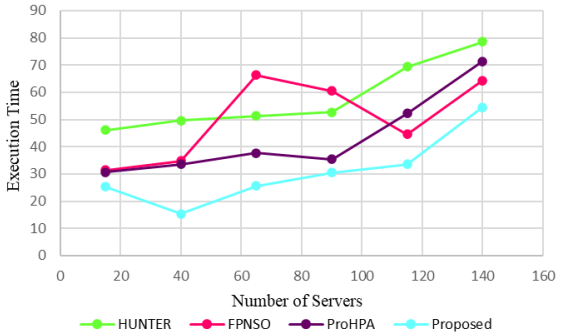


Figure. 7 Comparision in terms of execution time

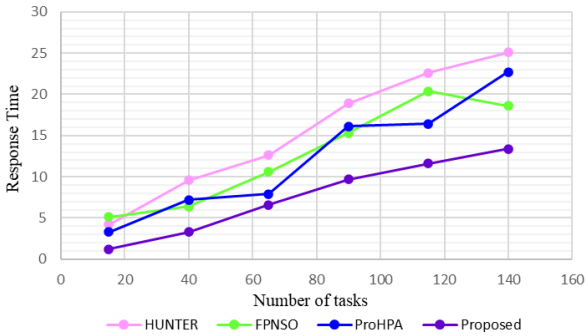
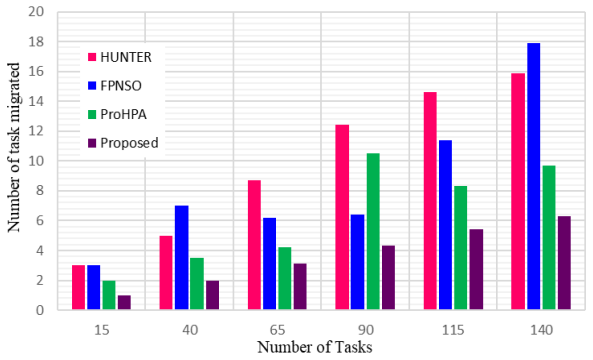
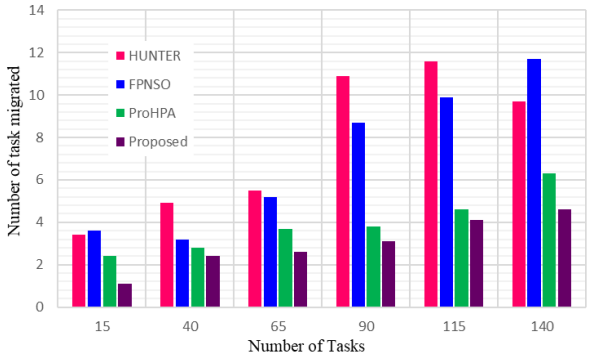


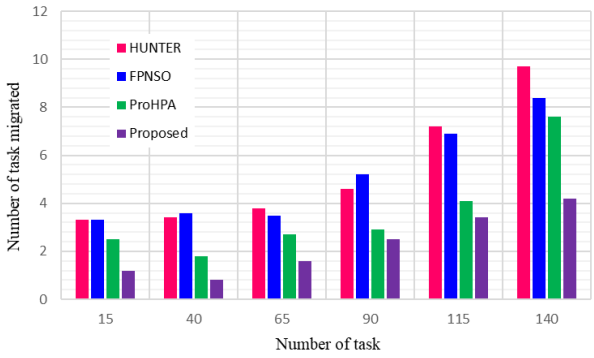
Figure. 8 Comparision in terms of response time



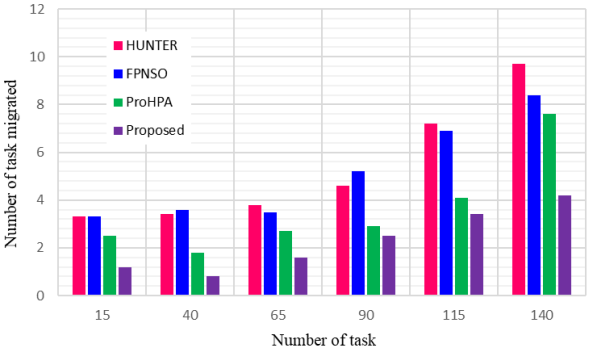
(a)



(b)



(c)



(d)

Figure. 9 Comparision in terms of task migration: (a) When amount of servers = 50, (b) When amount of servers = 100, (c) When amount of servers = 150, and (d) When amount of servers = 200

4.3.3. Performance based on task execution time

The task's execution time includes the time the system spends carrying out run-time or system functions on its behalf. The execution time is calculated using equation 8 as follows:

$$Ex_{Ti} = \sum_{y=1}^{N} L_{Ti}(L_y) \tag{9}$$

Where, $L_{Ti}(L_y)$ refers to the total time taken for execution of y^{th} task.

A comparison of the accomplishment times of our proposed CLUQOA technique with current HUNTER [14], FPNSO [16], and ProHPA [17] techniques, with different numbers of servers, is shown in Figure 7. Our approach exhibits improved efficiency in various server setups with consistently decreased execution times. The graph demonstrates how well our method works to expedite task completion, particularly as the number of servers rises.

4.3.4. Performance based on response time

The following formula in equation (7) is used to calculate response time.

$$\left[R_t = \sum_{y=1}^{N} C_{Tik} - \left(S_{i_y} - E_{i_y}\right)\right] \quad (10)$$

Where, y represents the task, R_t signifies the response time C_{Tik} signifies the computational time for task y, S_{i_y} represents the task start time of y[th] task and E_{i_y} represents the task end time of y[th] task.

A comparison of response times for different task numbers using our proposed CLUQOA technique with current HUNTER [14], FPNSO [16], and ProHPA [17], techniques are shown in Figure 8. Lower response times are a consistent result of our solution, demonstrating its higher effectiveness in handling user requests. The graph highlights how well our method works to optimize task scheduling for increased system responsiveness, particularly when job complexity fluctuates.

4.3.5. Performance based on tasks migrated

It is possible to move virtual machines (VMs) from one server to another so they can keep running. During task execution, this procedure—known as virtual machine migration—may take place more than once.

A thorough comparison of task migration across different task numbers using our proposed CLUQOA technique with current HUNTER [14], FPNSO [16], and ProHPA [17] techniques is shown in Figure 9. Our approach regularly shows fewer task migrations at server counts of 50, 100, 150, and 200, indicating improved resource allocation and stability. The graph demonstrates how scalable and effective our method is in reducing task migration disruptions, which is essential for preserving system stability. Our approach exhibits robustness as server capacity grows, highlighting its flexibility and efficiency in managing a range of workloads while reducing the requirement for task migrations, thereby improving system performance.

5. Conclusion

In this paper, a novel Cloud based Load balancing Using Quantum artificial bee colony Optimization Algorithm (CLUQOA) has been proposed for task scheduling that can significantly enhance the effectiveness of cloud computing operations. The proposed CLUQOA offers a unique method for load balancing in cloud systems by utilizing the ideas of quantum-artificial bee colony optimization. The algorithm's integration of quantum computing techniques improves variety and efficiency, which in turn leads to better resource distribution, dependability, and affordability. When compared to other methods like HUNTER, FPNSO, and ProHPA, CLUQOA was shown to be more effective after a thorough analysis of its performance that took into account measures like makespan, resource utilization, task execution time, response time, and job migration. The proposed method achieves better makespan of 11.02%, 9.6% and 10.4% than HUNTER, FPNSO, and ProHPA, techniques respectively. In the future, the CLUQOA technique may be expanded to handle dynamic and unpredictable workloads, its quantum-inspired components may be further optimized, and its relevance to new technologies like edge computing and hybrid cloud settings may be investigated.

Conflicts of Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Author Contributions

The following statements should be used as follows: “Conceptualization, G. Visalaxi and A. Muthukumaravel; methodology, G. Visalaxi; software, A. Muthukumaravel; validation, G. Visalaxi, and A. Muthukumaravel; formal analysis, G. Visalaxi; investigation, A. Muthukumaravel; resources, G. Visalaxi; data curation, A. Muthukumaravel; writing—original draft preparation, G. Visalaxi; writing—review and editing, A. Muthukumaravel; visualization, G. Visalaxi; supervision, A. Muthukumaravel; project administration, G. Visalaxi; funding acquisition, A. Muthukumaravel”, etc.

Acknowledgments

The author would like to express his heartfelt gratitude to the supervisor for his guidance and unwavering support during this research for his guidance and support.

References

[1] S.S. Gill, S. Tuli, M. Xu, I. Singh, K.V. Singh, D. Lindsay, S. Tuli, D. Smirnova, M. Singh, U. Jain, and H. Pervaiz, “Transformative effects of IoT, Blockchain and Artificial Intelligence on cloud computing: Evolution, vision, trends and open challenges”, *Internet of Things*, Vol. 8, pp.100118, 2019.

[2] N.A. Angel, D. Ravindran, P.D.R. Vincent, K. Srinivasan, and Y.C. Hu, “Recent advances in

evolving computing paradigms: Cloud, edge, and fog technologies", *Sensors*,Vol. 22, No. 1, pp.196, 2021.

[3] T. Khan, W. Tian, G. Zhou, S. Ilager, M. Gong, and R. Buyya, "Machine learning (ML)-centric resource management in cloud computing: A review and future directions", *Journal of Network and Computer Applications*, Vol. 204, pp.103405, 2022.

[4] A. Mohamed, M. Hamdan, S. Khan, A. Abdelaziz, S.F. Babiker, M. Imran, and M.N. Marsono, "Software-defined networks for resource allocation in cloud computing: A survey", *Computer Networks*, Vol. 195, pp.108151, 2021.

[5] V. Sathiyamoorthi, P. Keerthika, P. Suresh, Z.J. Zhang, A.P. Rao, and K. Logeswaran, "Adaptive fault tolerant resource allocation scheme for cloud computing environments", *Journal of Organizational and End User Computing (JOEUC)*, Vol. 33, No. 5, pp.135-152, 2021.

[6] E.H. Houssein, A.G. Gad, Y.M. Wazery, and P.N. Suganthan, "Task scheduling in cloud computing based on meta-heuristics: review, taxonomy, open challenges, and future trends", *Swarm and Evolutionary Computation*, Vol. 62, pp.100841, 2021.

[7] G. Liu, Z. Xiao, G. Tan, K. Li, and A.T. Chronopoulos, "Game theory-based optimization of distributed idle computing resources in cloud environments", *Theoretical Computer Science*, Vol. 806, pp.468-488, 2020.

[8] J. Praveenchandar, and A. Tamilarasi, "Dynamic resource allocation with optimized task scheduling and improved power management in cloud computing", *Journal of Ambient Intelligence and Humanized Computing*, Vol. 12, pp.4147-4159, 2021.

[9] M.A. Shahid, N. Islam, M.M. Alam, M.M. Su'ud, and S. Musa, "A comprehensive study of load balancing approaches in the cloud computing environment and a novel fault tolerance approach", *IEEE Access*, Vol. 8, pp.130500-130526, 2020.

[10] S.K. Mishra, B. Sahoo, and P.P. Parida, "Load balancing in cloud computing: a big picture", *Journal of King Saud University-Computer and Information Sciences*, Vol. 32, No. 2, pp.149-158, 2020.

[11] S.A. Murad, Z.R.M. Azmi, A.J.M. Muzahid, M.K.B. Bhuiyan, M. Saib, N. Rahimi, N.J. Prottasha, and A.K. Bairagi, "SG-PBFS: Shortest Gap-Priority Based Fair Scheduling technique for job scheduling in cloud environment", *Future Generation Computer Systems*, Vol. 150, pp.232-242, 2024.

[12] I.A. Alimi, R.K. Patel, N.J. Muga, A.N. Pinto, A.L. Teixeira, and P.P. Monteiro, "Towards enhanced mobile broadband communications: A tutorial on enabling technologies, design considerations, and prospects of 5G and beyond fixed wireless access networks", *Applied Sciences*, Vol. 11, No. 21, pp.10427, 2021.

[13] X. Yuan, H. Wang, Y. Yuan, and S. Zhang, "Design of an Intelligent Decision Model for Power Grid Fault Location and Isolation based on Topology Analysis", *International Journal of Thermo fluids*, pp.100536, 2023.

[14] S. Tuli, S.S. Gill, M. Xu, P. Garraghan, R. Bahsoon, S. Dustdar, R. Sakellariou, O. Rana, R. Buyya, G. Casale, and N.R. Jennings, "HUNTER: AI based holistic resource management for sustainable cloud computing", *Journal of Systems and Software*, Vol. 184, pp.111124, 2022.

[15] D. Saxena, I. Gupta, A.K. Singh, and C.N. Lee, "A fault tolerant elastic resource management framework toward high availability of cloud services", *IEEE Transactions on Network and Service Management*, Vol. 19, No. 3, pp.3048-3061, 2022.

[16] A.K. Singh, S.R. Swain, D. Saxena, and C.N. Lee, "A bio-inspired virtual machine placement toward sustainable cloud resource management", *IEEE Systems Journal*, 2023.

[17] B. Jeong, S. Baek, S. Park, J. Jeon, and Y.S. Jeong, "Stable and efficient resource management using deep neural network on cloud computing", *Neurocomputing*, Vol. 521, pp. 99-112, 2023.

[18] M. Shuaib, S. Bhatia, S. Alam, R.K. Masih, N. Alqahtani, S. Basheer, and M.S. Alam, "An Optimized, Dynamic, and Efficient Load-Balancing Framework for Resource Management in the Internet of Things (IoT) Environment", *Electronics*, Vol. 12, No. 5, pp.1104, 2023.

[19] A.M. Senthil Kumar, K. Padmanaban, A.K. Velmurugan, X.S. Asha Shiny, and D.K. Anguraj, "A novel resource management framework in a cloud computing environment using hybrid cat swarm BAT (HCSBAT) algorithm", *Distributed and Parallel Databases*, Vol. 41, No. 1-2, pp.53-63, 2023.

[20] K. Surya, and V.M.A. Rajam, "Novel Approaches for Resource Management Across Edge Servers", *International Journal of Networked and Distributed Computing*, Vol. 11, No. 1, pp.20-30, 2023.