Journal of Parallel and Distributed Computing 191 (2024) 104920

Contents lists available at ScienceDirect

Journal of Parallel and Distributed Computing

journal homepage: www.elsevier.com/locate/jpdc

Cost-aware quantum-inspired genetic algorithm for workflow scheduling in hybrid clouds

Mehboob Hussain [a,*], Lian-Fu Wei [a], Amir Rehman [a], Muqadar Ali [a], Syed Muhammad Waqas [b], Fakhar Abbas [c]

[a] School of Information Science and Technology, Southwest Jiaotong University, Chengdu 611756, China
[b] School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an, 710049, China
[c] Centre for Trusted Internet and Community, National University of Singapore (NUS), Singapore

ARTICLE INFO

Keywords:
Cloud computing
Workflow scheduling
Cost optimization
Quantum-inspired genetic algorithm
Hybrid cloud

ABSTRACT

Cloud computing delivers a desirable environment for users to run their different kinds of applications in a cloud. Numerous of these applications (tasks), such as bioinformatics, astronomy, biodiversity, and image analysis, are deadline-sensitive. Such tasks must be properly allocated to virtual machines (VMs) to avoid deadline violations, and they should reduce their execution time and cost. Due to the contradictory environment, minimizing the application task's completion time and execution cost is extremely difficult. Thus, we propose a Cost-aware Quantum-inspired Genetic Algorithm (CQGA) to minimize the execution time and cost by meeting the deadline constraints. CQGA is motivated by quantum computing and genetic algorithm. It combines quantum operators (measure, interference, and rotation) with genetic operators (selection, crossover, and mutation). Quantum operators are used for better population diversity, quick convergence, time-saving, and robustness. Genetic operators help to produce new individuals, have good fitness values for individuals, and play a significant role in preserving the evolution quality of the population. In addition, CQGA used a quantum bit as a probabilistic representation because it has higher population diversity attributes than other representations. The simulation outcome exhibits that the proposed algorithm can obtain outstanding convergence performance and reduced maximum cost than benchmark algorithms.

1. Introduction

Cloud computing abbreviated clouds has increased promptly in the enterprise and research society for a long time. It became famous because of the latest developments in virtualization technique [22]. It provides customers with services over the Internet, minimizing the need for platform management, operating costs, maintenance costs, and ensuring scalability [64], [1], [20]. The cloud delivers services to customers and permits them to concentrate on their firms without wasting energy and assets on building and preserving hardware platforms [10], [51]. We can divide the cloud into three major types, Public cloud, Private cloud, and Hybrid cloud, according to availability of resources [27]. Public clouds offer a limited collection of on-request VMs and charged users based on the pay-as-you-go model, where customers only pay for the resources they use [4], [3]. In contrast, private clouds are free to use and have enhanced security and data control, as organizations or entities own and manage their infrastructure. However, large application tasks cannot be executed by the deadline because of restricted computational power [56]. Instead, a hybrid cloud (HC) is a sophisticated solution that integrates both public and private cloud resources. It offers a balanced approach, providing organizations or entities with the flexibility, scalability, cost optimization, and data control required for their evolving business needs. In the HC, resources from the public cloud are utilized to complete a specific task when there is a shortage of available resources in the private cloud. Thereby, ideally scheduling workflow tasks on the heterogeneous computing resources in the HC systems have attracted the research community [23]. However, scheduling tasks in the HC systems are still challenging to optimize execution time and costs. One critical issue in task scheduling in an HC is the heterogeneity of resources across different cloud environments. It becomes challenging to optimize the execution time and meeting deadlines when tasks require specific resources not available in all clouds. Another issue is the dynamic and unpredictable nature of workload and resource availability in an HC. It is crucial to efficiently allocate resources and prioritize tasks

* Corresponding author at: Southwest Jiaotong University, Chengdu 611756, China.
E-mail address: mehboobskd@swjtu.edu.cn (M. Hussain).

https://doi.org/10.1016/j.jpdc.2024.104920
Received 26 June 2023; Received in revised form 11 February 2024; Accepted 11 May 2024
Available online 17 May 2024

to meet the desired execution time and deadline constraints. Additionally, ensuring data privacy and security during task execution across multiple cloud environments and customers must carefully determine which tasks should be executed in the private or public cloud to reduce execution time and costs is also a critical concern in HC task scheduling [65]. That's why, there is a demand for efficient method for scheduling tasks in the hybrid cloud, which ideally assign the tasks to the heterogeneous resources as well as reduce the execution time and costs.

Recently, a lot of researchers developed many heuristic task scheduling techniques to fulfill the quality of service (QoS) needs of tasks [44], [45], that involve task deadline [65], [55], makespan [43], [26] and cost [40], [8], [5]. However, [44], [45] did not consider the dynamic nature of task characteristics and system conditions. Task requirements, such as resource demands and deadlines, may change over time, making it difficult to ensure consistent QoS. Additionally, the heterogeneity of resources and the varying capabilities of different computing nodes pose challenges in achieving the desired QoS levels. Authors [65], [55] investigate a heuristic for scheduling a task in a hybrid cloud under deadline constraints. They should have taken into account the heterogeneity and dynamic nature of resources of hybrid clouds. It is not simple to efficiently assign tasks to the best suitable heterogeneous resources in the hybrid cloud. Achieving an optimal balance between the objective function and meeting deadlines is a complex optimization problem. Researchers developed a task scheduling framework to minimize the makespan [43], [26] and costs [40,8,5]. The challenge faced in these papers is the NP-hard nature of the problem, which means finding an optimal solution becomes computationally expensive as the problem size increases. Developing efficient algorithms that can handle large-scale task scheduling instances makes it challenging. Another challenge is the uncertainty and variability in task execution times and resource availability. Most of the above papers focused on the public or hybrid cloud with independent or dependent tasks. A common representation of workflows is through a Directed Acyclic Graph (DAG), where tasks are represented as nodes and dependencies between tasks are represented as edges [62], [61]. Due to the interdependence of tasks in workflow scheduling, the improper allocation of workflow tasks on cloud resources can lead to an increase in the execution time and cost-decreased overall cloud performance, and the violation of QoS assurances required by various user tasks [63]. The algorithms mentioned above needed to be revised when dealing with the scheduling problems that arise in cloud data centers. These problems tend to be large-scale and complex, requiring more advanced solutions.

Nowadays, evolutionary algorithms (EAs) are popular for task scheduling, and they find solutions that are near the optimal solution. Still, they face significant challenges when scheduling tasks in clouds due to the rapid increase in complexity as the number of tasks and resources grows. However, it is important to note that the applicability of EAs for task scheduling depends on several factors, including the specific assumptions and scenarios considered, the optimization objective (local or global), the scale of the system, and the level of system dynamics [18]. Recently, several researchers have investigated modern meta-heuristic algorithms to discover the best solution. Such like the methods [59], [11] can minimize their search area and make sure that they run within the allowed execution time, thus providing a balance between their execution time and the ultimate optimal schedule. The PSO [65], [34], [28], GA [2], ACO [16] and CS [31] are their typical examples. These EAs work on a group of possible solutions, employing the rules of survival of the fittest to consistently obtain the best solution estimation. A new group of approximations is produced by choosing individuals based on their fitness rank in the problem area and reproducing them by applying variation operators in each generation of the evolutionary algorithms. This operation can guide the adaptation of populations of individuals more adapted to the environment than those who created them, as in natural adaptation. Characterizing the individual represents EAs, and the evaluation function represents the individual's fitness level, as well as population dynamics such as population size, mutation operator, parental selection, reproduction and inheritance, and survival competition methods. Those components should be designed properly to have a good balance between exploration and exploitation. Therefore, there is a need to investigate the representation and variation operators to characterize individuals efficiently, examine the search space with a small number of individuals, and quickly exploit global solutions in the search area. Thus, various ideas of quantum computing are assumed in the suggested EA for these purposes.

This work suggests a cost-aware quantum-inspired genetic algorithm (CQGA) to address the discussed issue above. The CQGA is based on the idea and fundamentals of quantum computing. Like the meta-heuristic algorithms, CQGA is also characterized by the representation of the individual, the evaluation function, and the population dynamics. CQGA combines the quantum operators (measure, interference, and rotation) with genetic operators (selection, crossover, and mutation). In addition, CQGA uses a quantum bit (q-bit) as a probabilistic representation rather than a numeric, symbolic or binary representation because the q-bit representation has better population diversity characteristics than other representations. The measure function, interference, and quantum rotation gate are used for better solutions, quick convergence, time-saving, and robustness. Genetic operators help have good fitness values for individuals and play a significant role in preserving the evolution quality of the population. We carry out extensive experiments to demonstrate the performance of the CQGA algorithm. The experiment outcomes disclose that the CQGA algorithm surpassed benchmark methods.

The remainder of this work is structured in this way. We discussed the literature studies briefly in the following section. Section 3 described and formalized our problem along with mathematical models. Section 4 gives a key concept and fundamentals of quantum computing and genetic algorithm and describes our CQGA algorithm in Section 5. Section 6 comprehensively assesses the efficiency of CQGA, succeeded by conclusions in Section 7.

2. Literature review

In the past few years, efficient task scheduling in HC has captivated comprehensive from the academic community and enterprise [13], [50]. Scheduling workflow tasks to heterogeneous resources in an HC environment while considering execution time, cost, and Quality of Service (QoS) requirements is NP-hard [38]. The majority of existing literature on scheduling workflow applications focuses on single-cloud environments [24], [57], [52], [35], [7] with independent tasks or precedence-constrained workflow applications [49], [37], [41], [65]. We present a brief analysis of earlier research on task scheduling below that takes into consideration elements such as execution time, cost, and several levels of QoS.

Su et al. [48] proposed a state-of-the-art, cost-effective task scheduling approach employing a heuristic method to achieve the lowest completion time and cost by dynamically assigning tasks to cheap VMs. The authors in [15] presented an innovative cost-aware independent task scheduling approach for the heterogeneous virtualized cloud applying two practical methods. The results indicate that the proposed algorithm outperformed other algorithms in terms of both execution cost and makespan, demonstrating its effectiveness. In reference [36], the authors developed a cost-aware workflow scheduling framework that utilized a non-linear programming procedure and a heuristic method in the virtualized cloud. This framework aimed to enhance the task assignment operation by considering cost factors. Authors argue the proposed model achieves greater cost performance and lower task execution time. They proposed another cost-aware workflow scheduling framework inspired by GA to reduce the execution cost only. The simulation findings evidence that their suggested solution lowered execution costs sufficiently [39]. In [12], authors introduce two novel quantum-inspired meta-heuristic techniques for bi-level thresholding. These methods utilize quantum computing principles like qubits and state superposition

to achieve parallelism and leverage the discrete nature of quantum systems. The proposed methods are compared to classical approaches and the quantum evolutionary algorithm (QEA). Results demonstrate that the proposed methods are more time-efficient and outperform the QEA in terms of performance. In [47], the authors proposed a cost-aware workflow scheduling algorithm that assigns tasks to VMs to minimize execution time and cost. They implemented the algorithm using Java and validated its effectiveness through experiments conducted with the CloudSim toolkit. Likewise, in [60], authors designed a cost-aware task scheduling algorithm that optimizes makespan in a dynamic virtualized cloud by considering task requirements during VM allocation. The test findings showed that the suggested method gained less cost and makespan. We discovered that the above majority of works pay attention to independent task scheduling in a single cloud in place of precedence-constrained workflow task scheduling in the HC.

Therefore, we explore the scientific works on workflow scheduling in the HC network. Such as, authors [54] suggested a scheduling approach for scheduling tasks in the HC to enhance profitability for private cloud providers. The experimental outcomes reveal that the suggested method achieves less execution cost and response time. M. Hussain et al. [25] developed an efficient task-scheduling algorithm named DCWS, which addresses task-scheduling challenges in the HC. It involves executing the utmost workflow tasks on users' private cloud within the given deadline. Unscheduled tasks are then sent to the public cloud for execution. The model ensures efficient scheduling of workflow tasks on the private cloud and communication channels, meeting task-precedence requirements and deadline constraints while minimizing costs. Additionally, they employed sub-deadline division and optimal virtual machine selection techniques to reduce task slack time, execution time, and monetary expenses. In [17], the authors developed a technique for scheduling and executing bag-of-tasks in multi-cloud environments. They also proposed a flexible resource assignment technique that improves cloud resource utilization based on remaining free time. This technique incorporates a set of cloud scheduling algorithms. In [58], the authors addressed workflow scheduling issues in the virtualized cloud by proposing a data-aware scheduling framework. This framework manages budget requirements, enhances resource utilization, and provides guidelines for large-scale workflow scheduling. In their work [65], the authors proposed a novel resource assignment scheme in the IaaS cloud using a meta-heuristic-based scheduling technique. Simulation results demonstrated improved scheduling solutions while maintaining user-level quality of service.

The literature studies can be categorized into various forms following the characteristics of the examined challenges, for instance task interdependence, deadline constraints, execution cost, resource utilization, and a HC was considered in the current studies. Though, they studied a composition of two or three characteristics. All we know is that the task scheduling issue under consideration with all characteristics has yet to be considered.

3. Problem description

3.1. Proposed model

Fig. 1 illustrates our proposed system architecture combining private and public cloud resources. The user owns a private cloud, expressing our assumption that the operational costs for the private cloud are not considered, given that these costs are already factored into the budget. When executing tasks on a private cloud, the main concern is meeting deadlines. However, if the workflow application is too large for the private cloud's limited capacity, resources from the public cloud are requested. The suggested study focuses on HC scheduling to meet the requirements of workflow applications. If any task cannot be completed on the private cloud within the deadline, resources will be rented from the public cloud. The task scheduling operation in a heterogeneous HC network is demonstrated in Fig. 1, primarily comprised of two tiers.

Table 1
Symbols.

Symbols	Definition
VMs	Virtual Machines
HC	Hybrid Cloud
QoS	Quality of service
NM	Network Monitor
TM	Task Manager
T	Tasks set
E	Set of edges
t_q	Overall strength of tasks in a workflow
t_{entry}	Parent task
t_{exit}	Child task
t_i	The i^{th} task
t_j	The j^{th} task
S_i	Computation burden of task t_i
PR_i	Predecessor of task t_i
SU_i	Successor of task t_i
$dt(t_i, t_j)$	Data transmission from t_i to t_j
W	Workflow application
D_W	Workflow deadline
CPU	Central Processing Unit
C_{unit}	Computing unit
C_{bw}	Communication unit
$R^{private}$	Private cloud VMs
r_l	The l^{th} VM in private cloud
r_n	Overall strength of VMs in private cloud
U^{public}	Public cloud VMs
u_k	The k^{th} VM in public cloud
u_m	Overall strength of VMs in public cloud
M^{public}	Rented public cloud VMs
ET_i	Execution time of t_i
$\xi_{j,k}$	Computing capability
TT	Transmission time
ST_i	Start time of t_i
FT_i	Finish time of t_i
$V_{j,k}$	Available duration of VM for schedule task
C_i	Execution cost of t_i
P_{uk}	Price of rented VM
C_w	Execution cost of a workflow
QC	Quantum chromosome
QP	Quantum population
P_{cr}	The crossover probability
P_m	The mutation probability

The users submitted their workflow tasks to the System Agent containing NM, TM, and Scheduler. NM supervises the exchange of information between nodes and keeps data about the running tasks. Furthermore, when the System Agent receives the workflow tasks, they are sorted according to the SDF (Shortest Deadline First) method. TM produces a sequence of task execution and dispatches them to the scheduler. Lastly, the scheduler allocates every workflow task to the suitable virtual machine in a manner that takes a shorter execution time and cost. The notations used in the problem formulation are given in Table 1.

3.2. Workflow application model

A workflow is typically formed as a Directed Acyclic Graph (DAG) $W = \{T, E\}$, where T represents the set of tasks $\{t_1, t_2, t_3, ..., t_q\}$ and E represent the set of directed edges $\{(t_i, t_j) | i < j\}$ demonstrate the dependencies among tasks, and dependency (t_i, t_j) characterizes that the t_j cannot be executed until the t_i has finished, which is usually described as the control flow. Fig. 2 [23] demonstrates an example of DAG with ten tasks. As we can see from this DAG, the t_i is the first task with no parent called the entry task t_{entry}, and the last task t_{10} which has no child called exit task t_{exit}. Every workflow task is related to the computation burden, represented by S_i. After completion of immediate predecessor PR_i of task t_i, an immediate successor SU_i of task t_i can be executed. If the task t_i and t_j do not execute on the same VM, there is a data transmission from t_i to t_j, denoted by $dt(t_i, t_j)$. All tasks in a workflow W must be completed under the defined deadline constraints D_W.

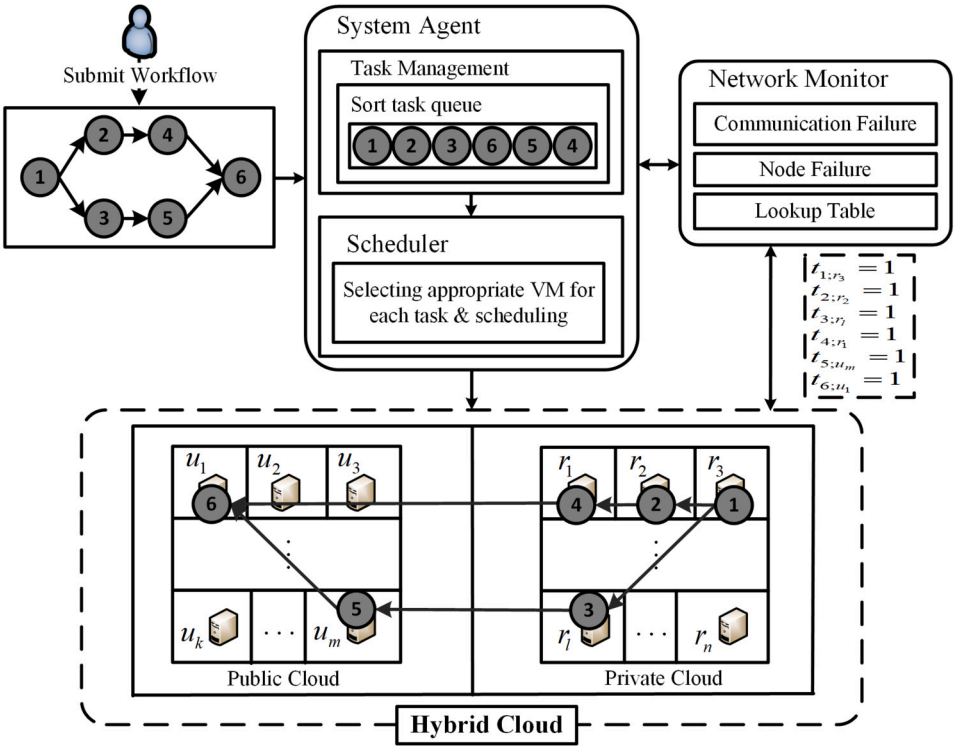


Fig. 1. The proposed system architecture for HC network.

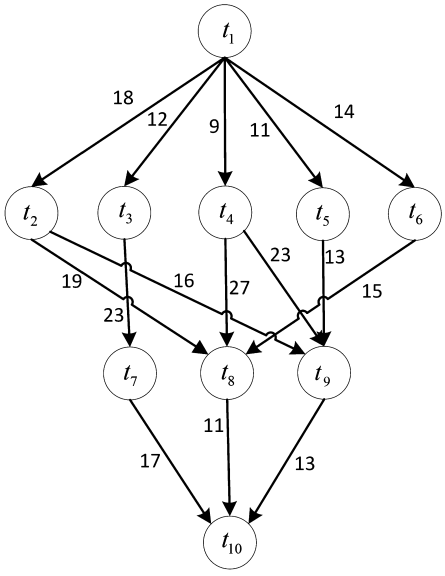


Fig. 2. A workflow instance.

3.3. Resource model

This study considers the CPU and bandwidth as the primary resources of the VM instance. We represent the computing unit of the CPU as C_{unit} and the communication bandwidth as C_{bw}. The private cloud computing resource (VM) is defined as follows:

$$R = \{C_{unit}, C_{bw}\} \tag{1}$$

Typically, the private cloud computing resources (VMs) are constrained and can be described as follows:

$$R^{private} = \{r_1, r_2, ..., r_l, ..., r_n\}, n \in N \tag{2}$$

In the given context, the notation $r_l (1 \le l \le n)$ represents the l-th VM, while n represents the total number of VMs owned by the private cloud provider.

Similarly, in the case of a public cloud, the VM is defined as:

$$U = \{C_{unit}, C_{bw}, price\} \tag{3}$$

where the term "price" refers to the monetary value set by public cloud providers for utilizing their resources.

The collection of free VMs in a public cloud is represented by U^{public}, which is defined as:

$$U^{public} = \{u_1, u_2, ..., u_k, ..., u_m\}, m \in N \tag{4}$$

where the notation $u_k (1 \le k \le m)$ represents the individual virtual machines (VMs), where k denotes the specific VM and m represents the total number of VMs available in the public cloud.

The HC pool consists of computing resources provided by both private and public clouds. While private cloud resources are available at no cost, the utilization of public cloud resources requires renting them. In the case of a large-size workflow application that cannot execute all workflow tasks on the private cloud within the given deadline, a public cloud VMs M^{public} take on rent to meet the workflow application requirements. Therefore, the HC's total resource count is determined by:

$$HC = R^{private} \cup M^{public}, M^{public} \subset U^{public} \tag{5}$$

3.4. Makespan model

Makespan is the total processing time of all tasks in a workflow application. It is calculated by the execution time and communication time of all tasks and between those tasks. Particularly, the execution time of task i depicted as ET_i is predicted by the computation time of task i and the computing ability $\xi_{j,k}$ of the computing resource (VM) $V_{j,k}$. The ET_i can be formulated as:

$$ET_i = \frac{S_i}{\xi_{j,k}}, \qquad V_{j,k} \in P^{private} \cup P^{public} \tag{6}$$

where S_i depicts the computation burden of workflow task t_i and $\xi_{j,k}$ depicts the computing ability of VM $V_{j,k}$.

As shown in Fig. 2, tasks depend on each other; the child tasks t_j cannot execute until the parent task t_i has finished. After finishing the task t_i, it sends the required information to its child task t_j. Therefore, we must examine the communication time between tasks. If the parent tasks t_i and the child task t_j execute on the same computing resource, their communication time will be zero; otherwise, it estimates its bandwidth and the volume of the transferred information. Let's assume, tasks t_i and t_j are allocated to virtual machine $V_{j,k}$ and $V_{j,l}$, respectively. Then, the transmission time $TT(t_i, t_j)$ among tasks t_i and t_j can be estimated as:

$$TT(t_i,t_j) = \begin{cases} 0, & V_{j,k} = V_{j,l}, \\ \frac{dt(t_i,t_j)}{min\{C_b(V_{j,k}), C_b(V_{j,l})\}}, & V_{j,k} \neq V_{j,l}, \end{cases} \tag{7}$$

where $dt(t_i, t_j)$ depicts the amount of data transmit from t_i to t_j, $C_b(V_{j,k})$, $C_b(V_{j,l})$ depicts the communication bandwidths of $V_{j,k}$ and $V_{j,l}$.

Hence, we can define the total completion time of a workflow application by the start time ST and the finish time FT of all tasks in a workflow. The ST is calculated by the FT of its predecessor task, TT between its predecessors, and the task completion time already processing on the same computing resource. Thereby, the ST_i of task t_i executing on a computing resource $V_{j,k}$ is estimated as:

$$ST_i = \begin{cases} 0, & \text{if } t_i = t_{entry} \\ \max\{avail(V_{j,k})\}, & \\ \max\limits_{t_i \in PR_i} \left\{ FT_{t_j} + TT(t_i, t_j) \right\} & \end{cases} \tag{8}$$

where $avail(V_{j,k})$ is the duration whereupon $V_{j,k}$ is available for schedule task. PR_i is the immediate predecessor of task t_i, FT_j is the finish time of task t_j and $TT(t_i, t_j)$ is the transmission time between task t_i and t_j.

Hence, we can easily estimate the finish time FT_i of task t_i is formulated as:

$$FT_i = ST_i + TT_i \tag{9}$$

Thus, we can calculate the total completion time called the makespan of a workflow application by

$$MS = \max_{t_i \in T} \left\{ FT_i \right\} \tag{10}$$

3.5. Cost model

The usage of VMs in a public cloud incurs charges, as they are not available for free. Consequently, we adopt a charging model where VMs are charged based on the duration of their utilization, measured in seconds. Thus, we can evaluate the execution cost C_i of task t_i on a computing resource by its execution time and rent cost of the computing resource. It is defined as:

$$C_i = (FT_i - ST_i) \times P_{uk} \tag{11}$$

where P_{uk} is rented cost of public cloud VM that execute task t_i.

Hence, the total execution cost of all tasks in a workflow application W is estimated by

$$C_W = \sum_{i=1}^{q} (FT_i - ST_i) \times P_{uk} \tag{12}$$

3.6. Problem model

The cost optimization problem for workflow is to reduce the execution cost of scheduling all tasks in a workflow W application satisfying the deadline constraint D_W. It can be expressed as:

$$Min : C_W \tag{13}$$

Subject to:

$$MS \leq D_W \tag{14}$$

where MS and C_W are presented in Equations: (10) and (12). To deal with the problem, we design a cost-aware quantum-inspired genetic algorithm called CQGA for workflow scheduling in HC.

4. Quantum genetic algorithm (QGA)

This part introduces the primary ideas and rules of quantum computing, which is vital to know the CQGA algorithm. The QGA merges quantum computing techniques and genetic algorithm (GA). GA produces new individuals through its variation operators (selection, crossover, and mutation). These variation operators help have good fitness values for individuals and play a significant duty in preserving the evolution quality of the population [53]. For this reason, many researchers used it in their optimization problems [23], [32], [30]. However, it has some drawbacks, including being time-consuming, slow convergence, and entangled in local optima [14], [6]. Therefore, we are developing a hybridized QGA algorithm by employing quantum computing techniques such as quantum bit (q-bit), quantum superposition, and quantum entanglement. Quantum modeling can be beneficial in finding good solutions in a large search space due to its unique properties. Quantum algorithms, such as quantum annealing or quantum-inspired algorithms, leverage quantum principles like superposition and entanglement to explore multiple potential solutions simultaneously. This allows for a more efficient exploration of the search space compared to classical algorithms. The reasoning behind this lies in the ability of quantum systems to explore and exploit the search space in a parallel and probabilistic manner. By representing problem solutions as quantum states, quantum modeling enables the exploration of multiple potential solutions simultaneously. This parallelism increases the chances of finding good solutions, especially in complex optimization problems with a large search space.

When it comes to scheduling problems, quantum modeling can help in moving towards local or global minima by leveraging quantum annealing or quantum-inspired algorithms. These algorithms utilize quantum principles to guide the search towards optimal or near-optimal solutions. By exploring the solution space in a quantum-mechanical way, these algorithms can potentially overcome the limitations of classical optimization methods and find better scheduling solutions. It's important to note that while quantum modeling shows promise in certain domains, it is still an area of active research and development. The effectiveness of quantum algorithms in solving specific problems depends on various factors, including problem complexity, available hardware, and algorithm design. Furthermore, we adopt a quantum rotation gate (q-gate) for quick convergence, time-saving, little population scale, and robustness [29].

4.1. Quantum computing concepts

In classical computing, a piece of information is stored in a binary number, either in state 0 or 1. In contrast, quantum computing is stored in a qubit called a quantum bit, either in state 1, 0 or a superposition of both [18]. It can be defined by

$$|\Psi> = \alpha|0> + \beta|1> \tag{15}$$

where α and β are complex numbers satisfying:

$$|\alpha|^2 + |\beta|^2 = 1 \tag{16}$$

where $\alpha|^2$ and $|\beta|^2$ depict a probability that a qubit is in state 0 or 1.

The population refines by changing the state of a qubit through the quantum gate. For this purpose, various quantum gates are available, including rotation gate, Hadamard gate, Not gate, controlled-NOT gate, etc. [14]. A quantum register of n qubits can define 2^n values at the same time. However, when the "measure" is taken, it collapses to a single state. This method enables an individual's amplitude to be altered to enhance performance, and it helps to lead each qubit toward the best solution [6].

$$\begin{bmatrix} \alpha' \\ \beta' \end{bmatrix} = \begin{bmatrix} cos(\vartheta) & -sin(\vartheta) \\ sin(\vartheta) & cos(\vartheta) \end{bmatrix} \times \begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix} \tag{17}$$

where α and β is modified to α' and β' using a single qubit rotation gate with an angle ϑ.

5. Proposed algorithm (CQGA)

This part introduces the proposed algorithm called CQGA to resolve workflow task scheduling issues in the HC network. The CQGA model comprises quantum computing techniques and a genetic algorithm. Quantum operators are used for quick convergence, time-saving, little population scale, and robustness, and genetic operators help have good fitness values for individuals and play a significant role in preserving the evolution quality of the population. The main aim of CQGA is to execute all tasks in the HC network in a manner that consumes the least completion time and execution cost under the given deadline. The complete procedure of the CQGA is demonstrated in Fig. 3 and Algorithm 1.

Algorithm 1: The CQGA Algorithm Framework.

```
begin
    t_q ← Total tasks in a workflow application
    p_j ← Total VMs in a hybrid cloud
    Sort all the tasks of a workflow
    Initialize quantum chromosome size according to the number of
      tasks in a workflow
    Generate initial quantum chromosome population
      QP(τ) = {QC_1(τ), QC_2(τ), ..., QC_{k-1}(τ), QC_k(τ)}, where QC_j^τ
      denotes j^th individual chromosome in the τ iteration
    Apply quantum measure function to convert each qubit to a bit and
      allocate to the appropriate processor
    Evaluate the valid population QP(τ)
    Select the best chromosome b and store it in b(τ)
    Perform quantum crossover, and mutation operations
    To get new population QP(τ + 1), Q gates are performed on qubits
      of the QP(τ) with b
    τ = τ + 1
    Terminated if the stopping condition met;
    Else:
    Go to step 7
    Till the scheduling criteria not meet for the b and total iteration not
      maximized
    The best chromosome b is scheduled on the corresponding VM
    Till all the tasks of the workflow not scheduled
```

5.1. Structure of quantum chromosome

The quantum chromosome is composed of a string of m qubits (superposition of all possible states), as shown in Fig. 4. Each chromosome has three genes to describe the quantum qubit superposition format. According to the researcher's attempts and experience, selecting genes with 40 bits can generate more acceptable results; therefore, a chromosome comprises 120 qubits. Genes containing more qubits could be related to superior partitioning around the space.

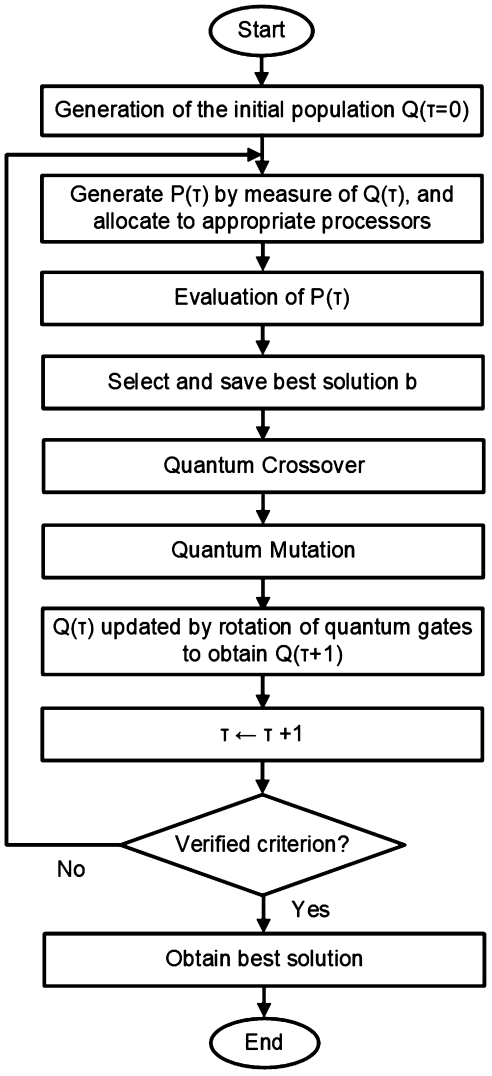


Fig. 3. The process of CQGA Algorithm.

In the quantum chromosome structure, the qubits can be used to encode the assignment of tasks to specific VM instances. Each qubit can represent a task and its corresponding assigned VM instance. The value or state of the qubit can indicate the specific VM instance type to which the task is assigned. By manipulating the quantum chromosome structure through quantum operations, such as quantum gates or rotations, the optimization algorithm can explore different combinations of task-to-VM assignments. The goal is to find an optimal or near-optimal assignment that minimizes the overall execution time and cost. Therefore, the relationship between the VM instance type and the task in the quantum chromosome structure is established through the encoding of the chromosome, where each qubit represents a task and its assigned VM instance type. The optimization algorithm then operates on this structure to find an optimal assignment that meets the desired objectives.

5.2. Population initialization

The quantum chromosome population is created by the position of the amplitude of all qubits to $\frac{1}{\sqrt{2}}$ [19]. i.e., all superposition states have equal probability in the initial population.

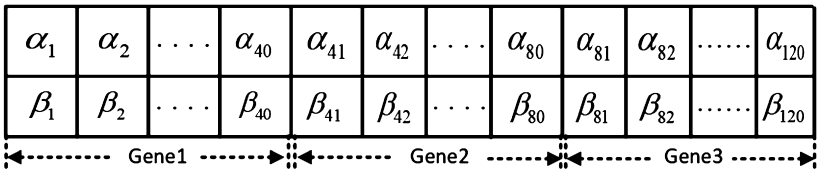


Fig. 4. Structure of quantum chromosome.

Table 2
Look-up table for Q-gate.

x_i	b_i	$f(x) > f(b)$	$\Delta\theta_i$	$s(\alpha_i, \beta_i)$			
				$\alpha_i\beta_i > 0$	$\alpha_i\beta_i < 0$	$\alpha_i = 0$	$\beta_i = 0$
0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	1	0	0	0	0	0	0
0	1	1	0.025	-1	+1	±1	0
1	0	0	0.01	-1	+1	±1	0
1	0	1	0.025	+1	-1	0	±1
1	1	0	0.02	+1	-1	0	±1
1	1	1	0.05	+1	-1	0	±1

5.3. Measuring chromosomes

The proposed algorithm is a stochastic algorithm that depends on the individual qubit length. We produce the initial quantum chromosome $QP(\tau) = \{QC_1(\tau), QC_2(\tau), ..., QC_{k-1}(\tau), QC_k(\tau)\}$ at τ iteration, where QC_j^{τ} denotes j^{th} individual consisting of qubits determined as:

$$QC_j(\tau) = \begin{bmatrix} \alpha_{j1}^{(\tau)} & \alpha_{j2}^{(\tau)} & \cdots & \alpha_{j(n-1)}^{(\tau)} & \alpha_{jn}^{(\tau)} \\ \beta_{j1}^{(\tau)} & \beta_{j2}^{(\tau)} & \cdots & \beta_{j(n-1)}^{(\tau)} & \beta_{jn}^{(\tau)} \end{bmatrix} \tag{18}$$

where n qubits depict virtual machines in the HC network, and employing quantum measure, the jth qubit changed into a classical binary bit. QC_j^0 comprise same possibility $\frac{1}{\sqrt{2}}$ for all qubits. An easy method to perform the quantum measurement function is shown in Algorithm 2, which randomly creates an integer r between 0 to 1. If the integer r is greater than $|\alpha_j|^2$, the value must be 1, else 0.

Algorithm 2: Procedure Q Measure.

```
begin
    j ← 1;
    while j≤n do
        if random[0,1] < |α_j|^2 then
            x_j ← 1;
        else
            x_j ← 0;
        j ← j + 1;
```

5.4. Task scheduling

The user submitted the set of tasks to the System Agent with their deadlines. All tasks are ordered in ascending ranking based on task deadlines. Then, to generate the initial population, we randomly pick one task from the queue. The total tasks q are defined according to the total tasks in a queue and the total computing resource unoccupied and represented as $q = m$. If this condition is not met, then q is the total tasks in a workflow and irregularly produces the size of each chromosome for allocating every task in $T = \{t_1, t_2, t_3, ..., t_q\}$ to a virtual machine in the HC network.

5.5. Chromosome evaluation and selection

Assessing the good and bad chromosomes in a population is necessary to achieve the desired goals. For this purpose, we use the fitness function, which helps us to know how much the chromosome "fit", "good", or "bad" relates to the issue under study. It also enables us to find the optimum solution and influence the convergence speed. Therefore, we developed the fitness function according to the issue under consideration, defined by

$$FT(\xi) = \begin{cases} C, MS \leqslant D_w \\ C + b \times (MS - D_w), MS > D_w, \end{cases} \tag{19}$$

where ξ depicts a chromosome and b is a constant that depicts a fine element. Suppose a makespan of a chromosome is not higher than a defined deadline. In that case, a cost only determines a fitness value; otherwise, an addition of cost and fine due to execution delay represents a fitness value. Once the fitness values of all chromosomes in a population are calculated, all population is arranged by their fitness values. After that, the proposed algorithm CQGA chooses a fixed proportion of chromosomes from the population with the lowest fitness values and discards the others.

5.6. Quantum crossover

We apply a quantum crossover operation to keep the population diversity. As reported by [12], we also set the crossover probability as $P_{cr} = 0.9$ to perform a single-point-crossover operation by choosing two chromosomes randomly at any random position. After applying a quantum crossover operation for each generation, a fresh chromosome pool is produced. This operation and its results are demonstrated in Fig. 5 while satisfying Equation (16) after applying this operator.

5.7. Quantum mutation

Once successfully applied, the crossover operator, then the mutation operator, is imposed to achieve exploration and exploitation. This method helps guarantee population diversity. Same as claimed by [12], we also set the mutation probability as $P_m = 0.1$, and every chromosome shall be mutated with other real numbers. Fig. 6 demonstrates this quantum mutation operation.

5.8. The interference

Interference enables the refinement of the population while altering the amplitudes of individuals to enhance performance. This operation completely works based on the current, best, and current solution amplitude signs. Each qubit of the quantum individual is revised by defining the rotation angle $U(\Delta\theta_i)$ and modifying the values using equations (20) and (21).

$$U(\Delta\theta_i) = \begin{bmatrix} cos(\Delta\theta_i) & -sin(\Delta\theta_i) \\ sin(\Delta\theta_i) & cos(\Delta\theta_i) \end{bmatrix} \tag{20}$$

$$\begin{bmatrix} \alpha_i^{\tau+1} \\ \beta_i^{\tau+1} \end{bmatrix} = \begin{bmatrix} cos(\Delta\theta_i) & -sin(\Delta\theta_i) \\ sin(\Delta\theta_i) & cos(\Delta\theta_i) \end{bmatrix} \times \begin{bmatrix} \alpha_i^{\tau} \\ \beta_i^{\tau} \end{bmatrix} \tag{21}$$

where $\Delta\theta_i$ depict a rotation angle of quantum gate i of each chromosome is generally taken from Table 2 to guarantee the convergence.

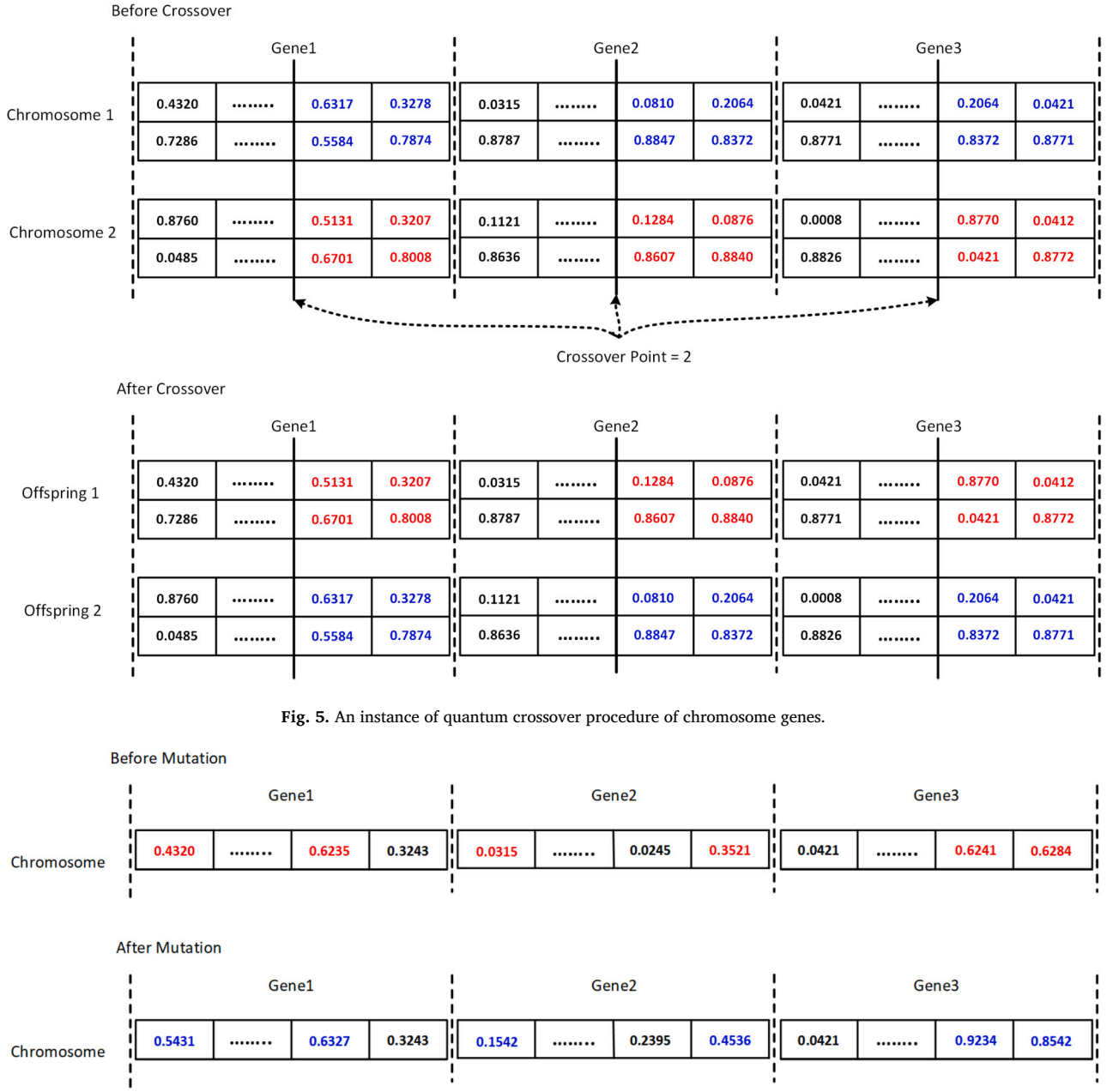


Fig. 5. An instance of quantum crossover procedure of chromosome genes.

Fig. 6. An instance of quantum mutation procedure of chromosome genes.

The rotation angle $\Delta\theta_i$ is determined by the magnitude and direction of rotation shown in Fig. 7. With the help of Table 2, and following its rules, $\Delta\theta_i$ is updated at each generation τ. The x_i and b_i given in the table depicts the i^{th} bits of x and b. Furthermore, $f(x)$ and $f(b)$ depicts the fitness function and $s(\alpha_i, \beta_i)$ is the sign of the rotation angle $\Delta\theta_i$. Following the Table 2, assume x_i value is 0 and b_i value is 1, $f(x_i)$ is greater than $f(b_i)$, this means that x_i is the current best solution. $\Delta\theta_i$ is modified according the $s(\alpha_i, \beta_i)$ given in the lookup table. At the final, $x(\tau)$ and $b(\tau - 1)$ are stored to $b(\tau)$, and then check verified criterion if the condition satisfied its stop; otherwise, produce a new population.

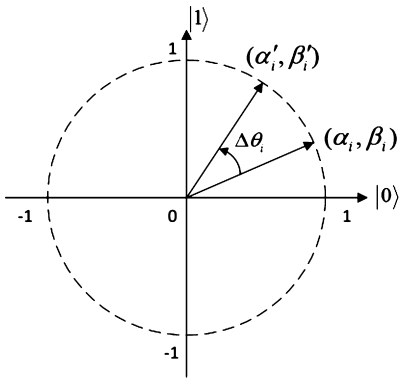


Fig. 7. Qubit conversion with quantum rotation gate.

6. Performance evaluation

We carry out a large-scale simulation test to check the efficiency progress obtained by the CQGA algorithm. Therefore, we compared it with four existing algorithms, including the classical genetic algorithm (CGA) [42], PSO [46], GWOA [21], and MDO-FF [33]. All algorithms

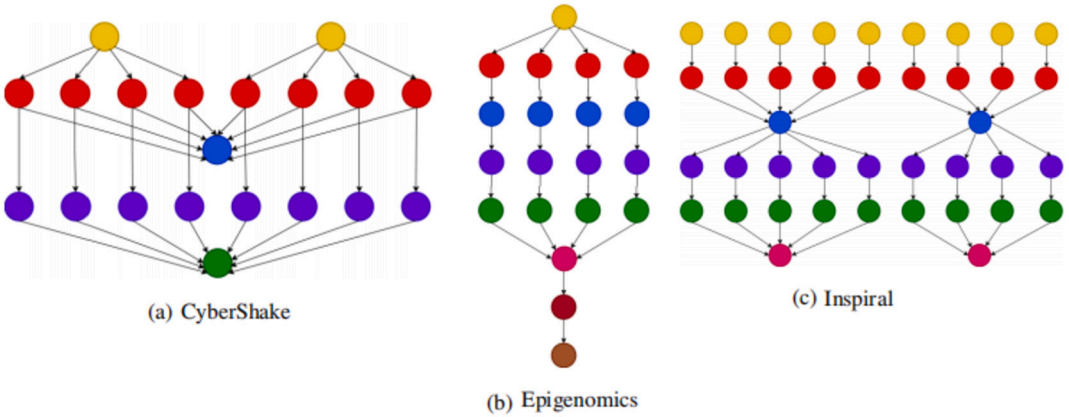


Fig. 8. The formation of three varieties of workflow.

Table 3
Attributes of three types of workflows.

Benchmarks	Nodes	Edges	Average data size (MB)	Average execution time (s)
CyberShake: 30/50/100	30/50/100	112/188/571	747.48/864.74/849.60	23.71/27.05/26.57
Epigenomics: 30/50/100	30/50/100	75/148/322	116.20/104.81/395.10	681.54/844.90/395.90
Inspiral: 30/50/100	30/50/100	95/160/319	9.00/9.16/8.93	206.78/226.19/206.12

Table 4
Deadline constraints for three varieties of workflows.

Workflow application	0.2MinTime	0.4MinTime	0.6MinTime	0.8MinTime	1MinTime
CyberShake-30	238.4	389.8	465.6	653.2	725.2
CyberShake-50	253.2	401.7	444.2	672.7	746.5
CyberShake-100	289.4	439.8	550.6	715.5	778.1
Epigenomics-30	226.2	298.5	366.7	562.4	599.4
Epigenomics-50	245.4	315.4	378.1	591.5	615.4
Epigenomics-100	295.6	384.8	425.7	643.4	688.6
Inspiral-30	452.3	598.6	667.2	893.6	958.5
Inspiral-50	471.4	621.5	698.8	915.1	993.9
Inspiral-100	509.8	655.7	780.3	966.4	999.2

are executed in the Python language and run on a Windows Server (Intel (R) Core (TM) i5-8265U CPU @ 1.60 GHz, 1.80 GHz 8 GB RAM).

6.1. Workload and platform

To validate the proposed algorithm CQGA, a comparison is carried out with four existing algorithms and three workflow applications, CyberShake, Epigenomics, and Inspiral, are used to verify the validity of the proposed algorithm under varying deadlines. The CQGA is assessed on different workflows as studied in [25]. These workflows are synthesized using the generator program given in [9]. It uses information gathered from the actual execution of workflows to generate a synthetic workflow. We used CyberShake (IO and network-intensive), Epigenomics (both compute-intensive and network-intensive), and Inspiral (compute-intensive) in the simulation. The workflow applications structure, characteristics (containing the number of nodes and edges as well as the averaged execution time and data size), and the deadline constraints for each workflow in the experiments are shown in Fig. 8, Table 3, and Table 4, receptively.

Table 5 and 6 show the setups for private and public VMs. The VM specifications of the private cloud include its types, computing speed, bandwidth, and total number of VMs. As the public cloud VM specifications include its types, computing speed, bandwidth, and price of VMs. Furthermore, we are using five workflow applications shown in Fig. 8 to prove the validity of CQGA while comparing it with four existing algorithms.

Table 5
VM specifications of public cloud.

VM Type	Computing speed (MIPS)	Bandwidth (BPS)	Price ($)
m1.small	1.70	39,321,600	0.06
m1.medium	3.75	85,196,800	0.12
m1.large	7.5	85,196,800	0.24
m1.xlarge	15	131,072,000	0.48
m3.medium	3.75	85,196,800	0.113
m3.large	7.50	85,196,800	0.225
m3.xlarge	15.00	131,072,000	0.45
m3.2xlarge	30.00	131,072,000	0.9

Table 6
VM specifications of private cloud.

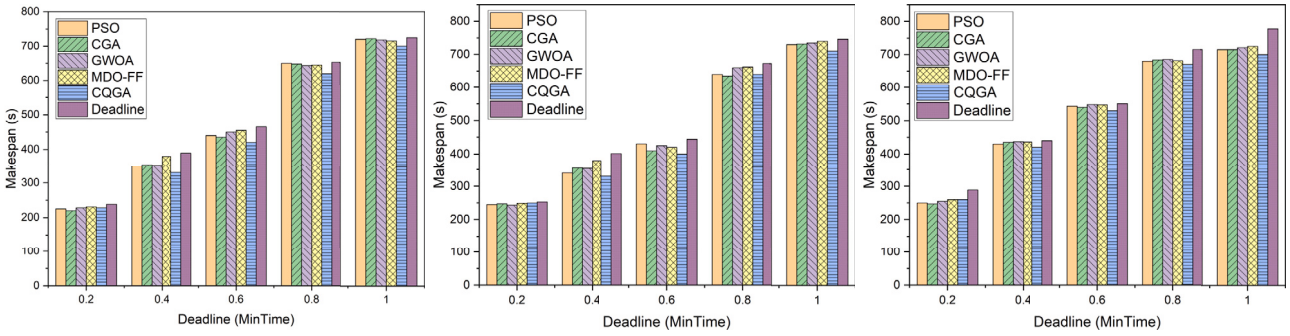
VM Type	Computing speed (MIPS)	Bandwidth (BPS)	Number
small	2	40,000,000	3
medium	3	60000,000	5
large	5	80,000,000	2

6.2. Deadline constraint

The deadline of a workflow application can be estimated as:

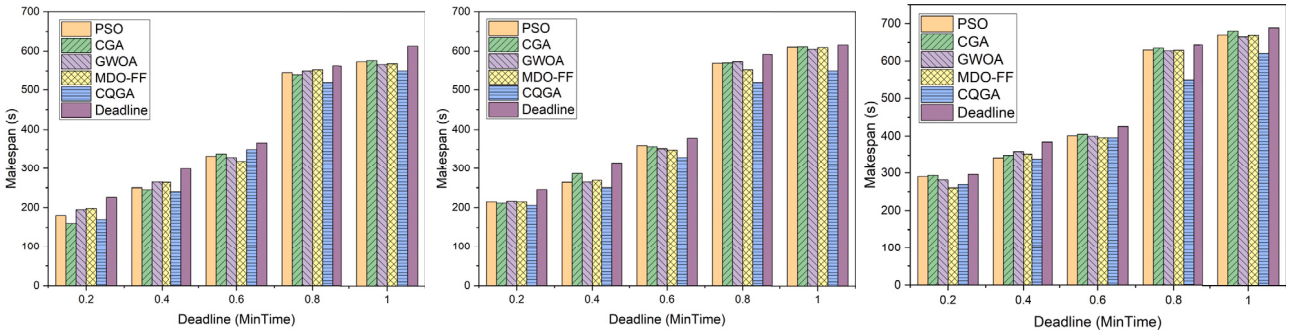
$$D_W = EFT_W + \gamma \times EFT_W \tag{22}$$

where γ depict the strictness of the deadline with the value range $\gamma \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$. EFT_W depict the earliest finish time of a workflow application.



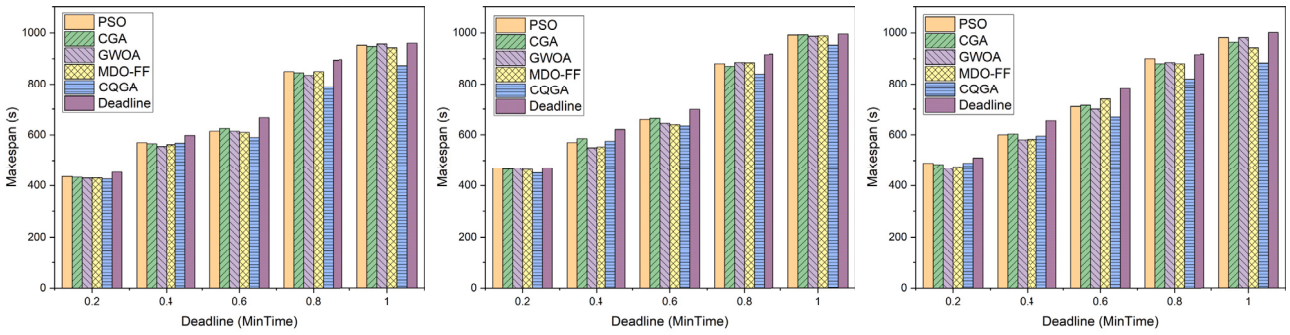
(a) CyberShake-30 (b) CyberShake-50 (c) CyberShake-100

Fig. 9. The makespan of CyberShake application under varying deadline constraints using different benchmark algorithms.



(a) Epigenomics-30 (b) Epigenomics-50 (c) Epigenomics-100

Fig. 10. The makespan of Epigenomic application under varying deadline constraints using different benchmark algorithms.



(a) Inspiral-30 (b) Inspiral-50 (c) Inspiral-100

Fig. 11. The makespan of Inspiral application under varying deadline constraints using different benchmark algorithms.

6.3. Results and discussion

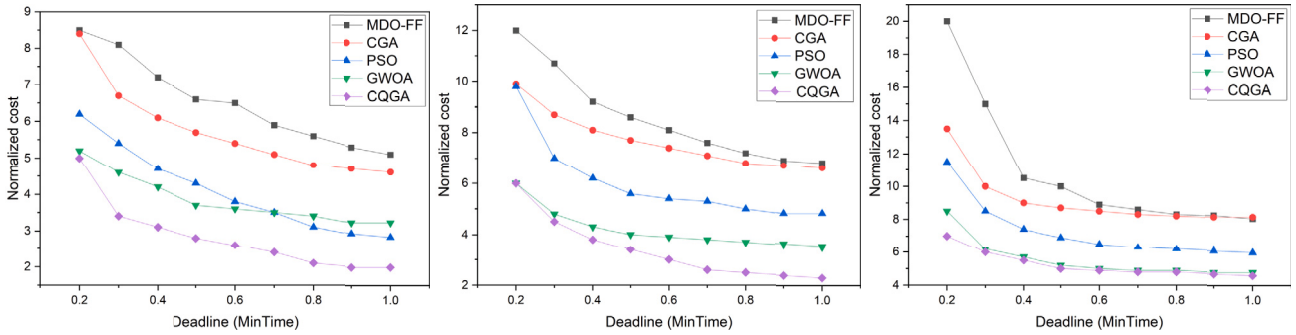
We investigate the testing outcomes of CQGA and other benchmark algorithms tested using CyberShake, Epigenomics, and Inspiral workflow applications. Fig. 9 - Fig. 11 represented the makespan of three different workflow applications under changing deadline restrictions employing PSO [46], CGA [42], GWOA [21], MDO-FF [33], and CQGA.

6.4. Makespan evaluation

In this part, we discuss the achieved makespan values of CyberShake, Epigenomic, and Inspiral workflow as depicted in Fig. 9 - Fig. 11. Each graph's line bar corresponds to the given interval's deadline value. By observing these values, we can evaluate which algorithm is superior to satisfy the deadline constraints. For the CyberShake, Epigenomic, and Inspiral workflows, all benchmark algorithms successfully execute the workflows under the defined deadlines.

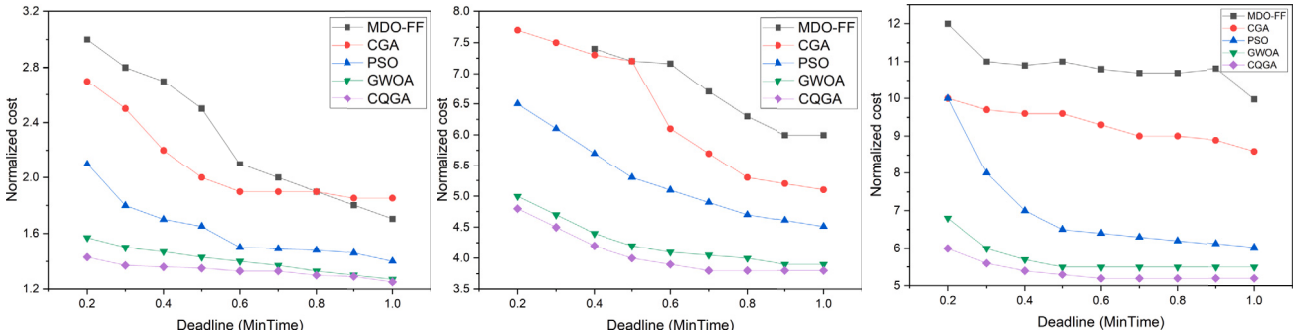
For the CyberShake workflow, we can observe from Fig. 9 that underneath the identical deadline constraints, the makespan achieved by PSO, CGA, GWOA, and MDO-FF initially is similar to that of CQGA and are close to a deadline. This means that they can generate schedules and complete them in due time. Moreover, CQGA achieved the lowest average makespan from 0.4 to 1 duration. The reason is that CQGA assumed a qubit to define the individual chromosome, which has more reasonable population variety features, and a genetic operator with a rotation gate to guide the schedule to better convergence. All benchmark algorithms can ensure the deadline constraints because all approaches focus on decreasing the execution time of workflow tasks.

In the Epigenomic and Inspiral workflow case, we can see from Fig. 10 and 11 when the number of workflows increases, the PSO and CGA have the higher average makespans while GWOA and MDO-FF have the lower on every case. This is because the maximum number of tasks in these workflow applications are parallel, which needs too much time to execute all tasks. In contrast, CQGA has more significant



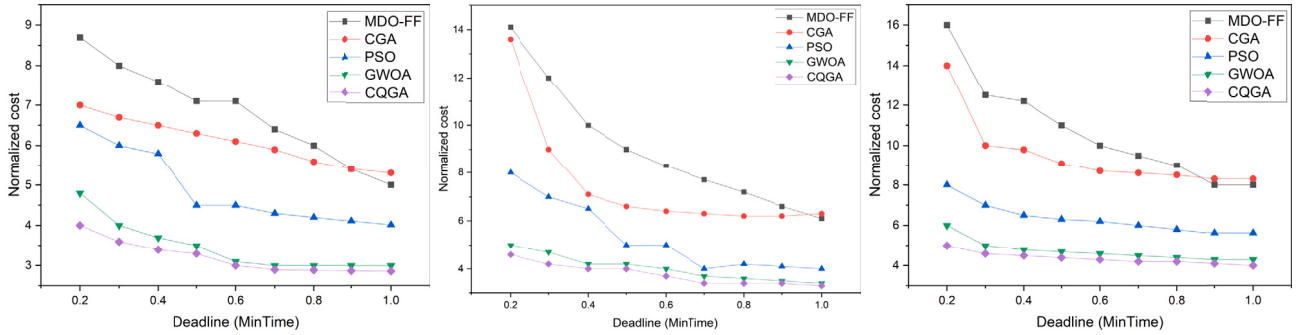
(a) CyberShake-30 (b) CyberShake-50 (c) CyberShake-100

Fig. 12. The normalized cost of CyberShake application under varying deadline constraints using different benchmark algorithms.



(a) Epigenomics-30 (b) Epigenomics-50 (c) Epigenomics-100

Fig. 13. The normalized cost of Epigenomic application under varying deadline constraints using different benchmark algorithms.



(a) Inspiral-30 (b) Inspiral-50 (c) Inspiral-100

Fig. 14. The normalized cost of Inspiral application under varying deadline constraints using different benchmark algorithms.

achievements than all existing algorithms. That is to say, CQGA exceeds existing algorithms regarding computational efficiency.

In conclusion, CQGA outperforms existing algorithms because of the HC network and quantum operators adopted in CQGA. The HC network enables CQGA to find schedules for workflow tasks to be finished on both the private and public cloud under the defined deadline. On the other hand, CQGA used quantum operators (measure, interference, and rotation gate), which have a better population diversity and lead the schedule to better convergence.

6.5. Cost evaluation

In this part, we evaluate the execution cost under varying deadline constraints using three benchmark algorithms as shown in Fig. 12 - Fig. 14. As illustrated in these figures, when the deadline is flexible, the execution cost of benchmark algorithms decreases for all three workflows. It is because the users desire to buy more inexpensive VMs with slower processing speeds and lower bandwidth to execute the workflow to reduce the execution cost under the flexible deadline. From these figures, we can discover that underneath the same deadline constraint, our proposed CQGA algorithm can considerably decrease execution cost compared to benchmark algorithms for CyberShake, Epigenomic, and Inspiral applications with changing nodes, respectively. These advantages are from the proposed HC network, quantum bits, quantum gates, and genetic operations adopted in our CQGA, which can discover a more satisfactory solution in exploring the entire solution area.

For the CyberShake workflow, the results for the MDO-FF and CGA algorithms exhibit an extensively higher cost comparing existing methods. In this particular scenario, MDO-FF and CGA have the lowest average makespan compared to the PSO and GWOA. In contrast, the remaining algorithms have less cost, however, at the price of taking longer execution times, which does not comply with the deadline constraints. The PSO and GWOA function the same in these specific cases and receive narrowly larger average makespans compared to MDO-FF and CGA despite considerably lower costs. By contrast, our proposed CQGA algorithm satisfies the deadline at each point by generating ef-

Table 7
ANOVA using Cybershake workflow of 1000 tasks.

(a) Summary of input

Group	Count	Sum	Average	Variance
CQGA	10	4704.984	470.4984	11156.2
GWOA	10	5906.008	590.6008	16514.14
PSO	10	6803.672	680.3672	26251.69
CGA	10	8802.784	880.2784	26233.35
MDO-FF	10	10302.67	1030.267	164442.8

(b) ANOVA test result

Source of Variation	SS	df	MS	F-statistical	p-value	F-critical
Between Groups	2019798.12455172	4	504949.53113793	10.32201981	5.12×10^{-6}	2.578739184
Within Groups	2201383.96452545	45	48919.6436561211			
Total	4221182	49				

Table 8
ANOVA using Epigenomics workflow of 1000 tasks.

(a) Summary of input

Group	Count	Sum	Average	Variance
CQGA	10	4704.8538	470.48538	4513.60863679068
GWOA	10	6204.534	620.4534	1780.95344444444
PSO	10	6704.9884	670.49884	17825.5304719693
CGA	10	9082.4983	908.24983	1079.66955469122
MDO-FF	10	10305.4298	1030.54298	4519.51081783289

(b) ANOVA test result

Source of Variation	SS	df	MS	F-statistical	p-value	F-critical
Between Groups	2044831.49109362	4	511207.872773406	86.00612034	1.75×10^{-20}	2.578739184
Within Groups	267473.456331557	45	5943.85458514571			
Total	2312304.94742518	49				

Table 9
ANOVA using Inspiral workflow of 1000 tasks.

(a) Summary of input

Group	Count	Sum	Average	Variance
CQGA	10	4102.6395	410.26395	994.186789578333
GWOA	10	4304.3439	430.43439	4526.67575957435
PSO	10	5903.9684	590.39684	9835.54437581139
CGA	10	9003.9398	900.39398	35490.1061960395
MDO-FF	10	10301.3899	1030.13899	64571.7453288366

(b) ANOVA test result

Source of Variation	SS	df	MS	F-statistical	p-value	F-cri
Between Groups	3139455.95610236	4	784863.989025591	34.00085911	4.49×10^{-13}	2.578739184
Within Groups	1038764.32604856	45	23083.6516899681			
Total	4178220.28215093	49				

fective schedules with lower makespan and costs. For deadline interval 0.2, CQGA needs to generate cheap schedules with tight deadlines. CQGA achieved minimum average cost for deadline durations 0.4, 0.6, 0.8, and 1.0, constructing it the best practical method by satisfying the deadlines at the minimum cost. In conclusion, CQGA outperforms all the other algorithms on intervals 0.4, 0.6, 0.8, and 1.0 by producing the cheapest and fastest schedule in most cases. Conversely, GWOA and CQGA fulfill average deadlines in each case, with CQGA generating the most efficient schedules with shorter makespans and lower prices due to free private cloud computing resources. If the number of tasks in a workflow is more significant and it is not possible to execute all those tasks on the private cloud, then CQGA requests a public cloud for a computing node; otherwise, CQGA tries to maximize the utilization of their private cloud computing resources. Furthermore, the quantum and genetic operators help to find more good solutions and guide the schedule to good convergence. As shown in Fig. 13 and Fig. 14, the obtained solutions of the Epigenomic and Inspiral workflow are all acceptable because of the large variety of solutions because of a low number of nodes and edges in these two applications, which is the cause of lower makespan time and execution cost.

Overall, we discovered that PSO and GWOA could produce low-cost schedules but fail to reduce the execution time in most cases. However, MDO-FF and CGA can produce schedules that satisfy the deadline constraints because their cost optimization method is better than others. In contrast, our CQGA produces a cheaper schedule and faster solutions. As expected, CQGA performs better than all other compared methods. The reason is that the techniques adopted in CQGA, such as quantum bit, measure, rotation gate, genetic operators, and the HC network, help to obtain exceptional achievement in discovering optimal results.

6.6. Analysis of variance (ANOVA) test

Here, we check the effectiveness of our proposed algorithm CQGA by using the well-known statistical test ANOVA [25]. In this test, we

match the means of the experimental results of our proposed algorithm with existing algorithms to evaluate whether there are considerable differences between them. That is to say, it helps in deciding even if the null hypothesis (H0), which indicates that the average of all algorithms is the same, can be rejected. For the purpose of rejecting the null hypothesis, the P-value is supposed to be smaller than the chosen α-level (= 0.05). In the same way, the value of the F-statistic is supposed to be greater than the F-critical. To achieve this goal, we describe a null hypothesis as:

$$H_0 : \text{CQGA} = \text{GWOA} = \text{PSO} = \text{CGA} = \text{MDO-FF} \quad (23)$$

Likewise, the alternative hypothesis can be described as:

$$H_1 : \text{Means are not equal} \quad (24)$$

The statistical test was carried out to check the comparison of the proposed algorithm CQGA, GWOA, PSO, CGA, and MDO-FF. To perform this test, all five algorithms were executed 10 times for each of the three workflow applications of different sizes. The results obtained from ANOVA test are demonstrated in Table 7, Table 8, and Table 9 for Cybershake, Epigenomic, and Inspiral workflows respectively. We find that, for all three workflows, we have F-statistical > F-critical. Hence, we can reject the null hypothesis. Therefore, the means of all the algorithms are significantly different. This implies that the performance of CQGA is surpassed and consistent with GWOA, PSO, CGA, and MDO-FF.

7. Conclusions

This work considered a workflow scheduling issue in an HC network. Therefore, we developed a quantum-inspired genetic algorithm called CQGA to resolve the issues under study. CQGA is a composite of quantum computing techniques and GA. Like the meta-heuristic algorithms, CQGA is also expressed by the representation of the individual, the evaluation function, and the population dynamics. It combines quantum operators with genetic operators. Quantum operators help to find better solutions, quick convergence, time-saving, and robustness. Genetic operators help to have good fitness values for individuals and play a significant role in preserving the evolution quality of the population. The algorithm comparison and result discussion section reveals that CQGA has superior scheduling performance to benchmark approaches.

However, it has certain limitations. For example, the current hybrid approach may not fully optimize resource utilization and struggle to adapt to dynamic environments where task characteristics and resource availability frequently change. Therefore, in our future work, we plan to integrate machine learning techniques into quantum-inspired algorithms to enhance adaptability and decision-making capabilities. By leveraging historical data and real-time feedback, these algorithms can dynamically adjust scheduling decisions to optimize performance in dynamic environments.

CRediT authorship contribution statement

Mehboob Hussain: Conceptualization, Data curation, Formal analysis, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Lian-Fu Wei:** Formal analysis, Funding acquisition, Project administration, Resources, Supervision. **Amir Rehman:** Writing – review & editing. **Muqadar Ali:** Writing – review & editing. **Syed Muhammad Waqas:** Writing – review & editing. **Fakhar Abbas:** Formal analysis, Software, Writing – review & editing.

Declaration of competing 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.

Acknowledgment

Funding for this work is provided by the National Key Research and Development Program of China (NKRDC) under Grant No. 2021YFA0718803 and Fundamental Research Funds for the Central Universities under Grant No. 2682024CX018.

References

[1] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, et al., A view of cloud computing, Commun. ACM 53 (4) (2010) 50–58.
[2] J.U. Arshed, M. Ahmed, T. Muhammad, M. Afzal, M. Arif, B. Bazezew, Ga-irace: genetic algorithm-based improved resource aware cost-efficient scheduler for cloud fog computing environment, Wirel. Commun. Mob. Comput. (2022) 2022.
[3] L.F. Bittencourt, E.R.M. Madeira, Hcoc: a cost optimization algorithm for workflow scheduling in hybrid clouds, J. Internet Serv. Appl. 2 (3) (2011) 207–227.
[4] R. Buyya, C.S. Yeo, S. Venugopal, J. Broberg, I. Brandic, Cloud computing and emerging it platforms: vision, hype, and reality for delivering computing as the 5th utility, Future Gener. Comput. Syst. 25 (6) (2009) 599–616.
[5] K.K. Chakravarthi, L. Shyamala, V. Vaidehi, Cost-effective workflow scheduling approach on cloud under deadline constraint using firefly algorithm, Appl. Intell. 51 (2021) 1629–1644.
[6] S. Chen, Z. Li, B. Yang, G. Rudolph, Quantum-inspired hyper-heuristics for energy-aware scheduling on heterogeneous computing systems, IEEE Trans. Parallel Distrib. Syst. 27 (6) (2015) 1796–1810.
[7] Y. Chen, G. Xie, R. Li, Reducing energy consumption with cost budget using available budget preassignment in heterogeneous cloud computing systems, IEEE Access 6 (2018) 20572–20583.
[8] F. Cheng, Y. Huang, B. Tanpure, P. Sawalani, L. Cheng, C. Liu, Cost-aware job scheduling for cloud instances using deep reinforcement learning, Clust. Comput. (2022) 1–13.
[9] A. Choudhary, I. Gupta, V. Singh, P.K. Jana, A gsa based hybrid algorithm for bi-objective workflow scheduling in cloud computing, Future Gener. Comput. Syst. 83 (2018) 14–26.
[10] P. Cong, L. Li, J. Zhou, K. Cao, T. Wei, M. Chen, S. Hu, Developing user perceived value based pricing models for cloud markets, IEEE Trans. Parallel Distrib. Syst. 29 (12) (2018) 2742–2756.
[11] C. Delimitrou, D. Sanchez, C. Kozyrakis, Tarcil: reconciling scheduling speed and quality in large shared clusters, in: Proceedings of the Sixth ACM Symposium on Cloud Computing, 2015, pp. 97–110.
[12] S. Dey, S. Bhattacharyya, U. Maulik, Quantum inspired genetic algorithm and particle swarm optimization using chaotic map model based interference for gray level image thresholding, Swarm Evol. Comput. 15 (2014) 38–57.
[13] J.J. Durillo, R. Prodan, Multi-objective workflow scheduling in Amazon ec2, Clust. Comput. 17 (2) (2014) 169–189.
[14] T. Gandhi, T. Alam, et al., Quantum genetic algorithm with rotation angle refinement for dependent task scheduling on distributed systems, in: 2017 Tenth International Conference on Contemporary Computing (IC3), IEEE, 2017, pp. 1–5.
[15] C. Gogos, C. Valouxis, P. Alefragis, G. Goulas, N. Voros, E. Housos, Scheduling independent tasks on heterogeneous processors using heuristics and column pricing, Future Gener. Comput. Syst. 60 (2016) 48–66.
[16] P. Gupta, U. Goyal, V. Verma, Cost-aware ant colony optimization for resource allocation in cloud infrastructure, Recent Adv. Comput. Sci. Commun., Former., Recent Patents Comput. Sci. 13 (3) (2020) 326–335.
[17] J.O. Gutierrez-Garcia, K.M. Sim, A family of heuristics for agent-based elastic cloud bag-of-tasks concurrent scheduling, Future Gener. Comput. Syst. 29 (7) (2013) 1682–1699.
[18] K.-H. Han, J.-H. Kim, Quantum-inspired evolutionary algorithm for a class of combinatorial optimization, IEEE Trans. Evol. Comput. 6 (6) (2002) 580–593.
[19] K.-H. Han, J.-H. Kim, Quantum-inspired evolutionary algorithms with a new termination criterion, h/sub/spl epsi//gate, and two-phase scheme, IEEE Trans. Evol. Comput. 8 (2) (2004) 156–169.
[20] H. He, G. Xu, S. Pang, Z. Zhao, Amts: adaptive multi-objective task scheduling strategy in cloud computing, China Commun. 13 (4) (2016) 162–171.
[21] F. Hemasian-Etefagh, F. Safi-Esfahani, Dynamic scheduling applying new population grouping of whales meta-heuristic in cloud computing, J. Supercomput. 75 (10) (2019) 6386–6450.
[22] M. Hussain, L.-F. Wei, A. Lakhan, S. Wali, S. Ali, A. Hussain, Energy and performance-efficient task scheduling in heterogeneous virtualized cloud computing, Sustain. Comput., Inf. Syst. (2021) 100517.
[23] M. Hussain, L.-F. Wei, F. Abbas, A. Rehman, M. Ali, A. Lakhan, A multi-objective quantum-inspired genetic algorithm for workflow healthcare application scheduling with hard and soft deadline constraints in hybrid clouds, Appl. Soft Comput. (2022) 109440.
[24] M. Hussain, L.-F. Wei, A. Rehman, F. Abbas, A. Hussain, M. Ali, Deadline-constrained energy-aware workflow scheduling in geographically distributed cloud data centers, Future Gener. Comput. Syst. 132 (2022) 211–222.

[25] M. Hussain, M.-X. Luo, A. Hussain, M.H. Javed, Z. Abbas, L.-F. Wei, Deadline-constrained cost-aware workflow scheduling in hybrid cloud, Simul. Model. Pract. Theory 129 (2023) 102819.
[26] N. Jain, I. Menache, J. Naor, J. Yaniv, Near-optimal scheduling mechanisms for deadline-sensitive jobs in large computing clusters, ACM Trans. Parallel Comput. (TOPC) 2 (1) (2015) 1–29.
[27] B. Javadi, J. Abawajy, R. Buyya, Failure-aware resource provisioning for hybrid cloud infrastructure, J. Parallel Distrib. Comput. 72 (10) (2012) 1318–1331.
[28] Q. Kang, M. Zhou, J. An, Q. Wu, Swarm intelligence approaches to optimal power flow problem with distributed generator failures in power networks, IEEE Trans. Autom. Sci. Eng. 10 (2) (2012) 343–353.
[29] D. Konar, S. Bhattacharyya, K. Sharma, S. Sharma, S.R. Pradhan, An improved hybrid quantum-inspired genetic algorithm (hqiga) for scheduling of real-time task in multiprocessor system, Appl. Soft Comput. 53 (2017) 296–307.
[30] D. Konar, K. Sharma, V. Sarogi, S. Bhattacharyya, A multi-objective quantum-inspired genetic algorithm (mo-qiga) for real-time tasks scheduling in multiprocessor environment, Proc. Comput. Sci. 131 (2018) 591–599.
[31] P. Krishnadoss, P. Jacob, Oloa: based task scheduling in heterogeneous clouds, Int. J. Intell. Eng. Syst. 12 (1) (2019) 114–122.
[32] R. Lahoz-Beltra, Quantum genetic algorithms for computer scientists, Computers 5 (4) (2016) 24.
[33] H. Li, Y. Wang, J. Huang, Y. Fan, Mutation and dynamic objective-based farmland fertility algorithm for workflow scheduling in the cloud, J. Parallel Distrib. Comput. 164 (2022) 69–82.
[34] J. Li, J. Zhang, C. Jiang, M. Zhou, Composite particle swarm optimizer with historical memory for function optimization, IEEE Trans. Cybern. 45 (10) (2015) 2350–2363.
[35] B. Lin, W. Guo, G. Chen, N. Xiong, R. Li, Cost-driven scheduling for deadline-constrained workflow on multi-clouds, in: 2015 IEEE International Parallel and Distributed Processing Symposium Workshop, IEEE, 2015, pp. 1191–1198.
[36] W. Lin, C. Liang, J.Z. Wang, R. Buyya, Bandwidth-aware divisible task scheduling for cloud computing, Softw. Pract. Exp. 44 (2) (2014) 163–174.
[37] F. Liu, B. Luo, Y. Niu, Cost-effective service provisioning for hybrid cloud applications, Mob. Netw. Appl. 22 (2) (2017) 153–160.
[38] J. Liu, L. Pineda, E. Pacitti, A. Costan, P. Valduriez, G. Antoniu, M. Mattoso, Efficient scheduling of scientific workflows using hot metadata in a multisite cloud, IEEE Trans. Knowl. Data Eng. 31 (10) (2018) 1940–1953.
[39] L. Liu, M. Zhang, R. Buyya, Q. Fan, Deadline-constrained coevolutionary genetic algorithm for scientific workflow scheduling in cloud computing, Concurr. Comput., Pract. Exp. 29 (5) (2017) e3942.
[40] S. Long, W. Long, Z. Li, K. Li, Y. Xia, Z. Tang, A game-based approach for cost-aware task assignment with qos constraint in collaborative edge and cloud environments, IEEE Trans. Parallel Distrib. Syst. 32 (7) (2020) 1629–1640.
[41] M. Malawski, K. Figiela, J. Nabrzyski, Cost minimization for computational applications on hybrid cloud infrastructures, Future Gener. Comput. Syst. 29 (7) (2013) 1786–1794.
[42] J. Meena, M. Kumar, M. Vardhan, Cost effective genetic algorithm for workflow scheduling in cloud under deadline constraint, IEEE Access 4 (2016) 5065–5082.
[43] S.K. Panda, P.K. Jana, Efficient task scheduling algorithms for heterogeneous multi-cloud environment, J. Supercomput. 71 (4) (2015) 1505–1533.
[44] S. Potluri, K.S. Rao, Optimization model for qos based task scheduling in cloud computing environment, Indones. J. Electr. Eng. Comput. Sci. 18 (2) (2020) 1081–1088.
[45] M.A. Rakrouki, N. Alharbe, Qos-aware algorithm based on task flow scheduling in cloud computing environment, Sensors 22 (7) (2022) 2632.
[46] M.A. Rodriguez, R. Buyya, Deadline based resource provisioning and scheduling algorithm for scientific workflows on clouds, IEEE Trans. Cloud Comput. 2 (2) (2014) 222–235.
[47] K. Sreenu, M. Sreelatha, W-scheduler: whale optimization for task scheduling in cloud computing, Clust. Comput. 22 (1) (2019) 1087–1098.
[48] S. Su, J. Li, Q. Huang, X. Huang, K. Shuang, J. Wang, Cost-efficient task scheduling for executing large programs in the cloud, Parallel Comput. 39 (4–5) (2013) 177–188.
[49] C. Vecchiola, R.N. Calheiros, D. Karunamoorthy, R. Buyya, Deadline-driven provisioning of resources for scientific applications in hybrid clouds with aneka, Future Gener. Comput. Syst. 28 (1) (2012) 58–65.
[50] A. Verma, S. Kaushal, A hybrid multi-objective particle swarm optimization for scientific workflow scheduling, Parallel Comput. 62 (2017) 1–19.
[51] X. Wang, L.T. Yang, L. Kuang, X. Liu, Q. Zhang, M.J. Deen, A tensor-based big-data-driven routing recommendation approach for heterogeneous networks, IEEE Netw. 33 (1) (2019) 64–69.
[52] Y. Wang, W. Shi, Budget-driven scheduling algorithms for batches of mapreduce jobs in heterogeneous clouds, IEEE Trans. Cloud Comput. 2 (3) (2014) 306–319.
[53] X. You, X. Miao, S. Liu, Quantum computing-based ant colony optimization algorithm for tsp, in: 2009 2nd International Conference on Power Electronics and Intelligent Transportation System (PEITS), vol. 3, IEEE, 2009, pp. 359–362.
[54] H. Yuan, J. Bi, W. Tan, B.H. Li, Temporal task scheduling with constrained service delay for profit maximization in hybrid clouds, IEEE Trans. Autom. Sci. Eng. 14 (1) (2016) 337–348.
[55] H. Yuan, J. Bi, W. Tan, M. Zhou, B.H. Li, J. Li, Ttsa: an effective scheduling approach for delay bounded tasks in hybrid clouds, IEEE Trans. Cybern. 47 (11) (2016) 3658–3668.
[56] H. Yuan, J. Bi, M. Zhou, Temporal task scheduling of multiple delay-constrained applications in green hybrid cloud, IEEE Trans. Serv. Comput. 14 (5) (2018) 1558–1570.
[57] L. Zeng, B. Veeravalli, X. Li, Scalestar: budget conscious scheduling precedence-constrained many-task workflow applications in cloud, in: 2012 IEEE 26th International Conference on Advanced Information Networking and Applications, IEEE, 2012, pp. 534–541.
[58] L. Zeng, B. Veeravalli, A.Y. Zomaya, An integrated task computation and data management scheduling strategy for workflow applications in cloud environments, J. Netw. Comput. Appl. 50 (2015) 39–48.
[59] Z.-H. Zhan, X.-F. Liu, Y.-J. Gong, J. Zhang, H.S.-H. Chung, Y. Li, Cloud computing resource scheduling and a survey of its evolutionary approaches, ACM Comput. Surv. 47 (4) (2015) 1–33.
[60] P. Zhang, M. Zhou, Dynamic cloud task scheduling based on a two-stage strategy, IEEE Trans. Autom. Sci. Eng. 15 (2) (2017) 772–783.
[61] Q. Zhao, Z. Gu, H. Zeng, N. Zheng, Schedulability analysis and stack size minimization with preemption thresholds and mixed-criticality scheduling, J. Syst. Archit. 83 (2018) 57–74.
[62] J. Zhou, J. Yan, K. Cao, Y. Tan, T. Wei, M. Chen, G. Zhang, X. Chen, S. Hu, Thermal-aware correlated two-level scheduling of real-time tasks with reduced processor energy on heterogeneous mpsocs, J. Syst. Archit. 82 (2018) 1–11.
[63] X. Zhou, G. Zhang, J. Sun, J. Zhou, T. Wei, S. Hu, Minimizing cost and makespan for workflow scheduling in cloud using fuzzy dominance sort based heft, Future Gener. Comput. Syst. 93 (2019) 278–289.
[64] X. Zhu, M. Hussain, X. Li, Energy-efficient independent task scheduling in cloud computing, in: International Conference on Human Centered Computing, Springer, 2018, pp. 428–439.
[65] X. Zuo, G. Zhang, W. Tan, Self-adaptive learning pso-based deadline constrained task scheduling for hybrid iaas cloud, IEEE Trans. Autom. Sci. Eng. 11 (2) (2013) 564–573.

Mehboob Hussain received a Ph.D. degree in Computer Science and Technology from the School of Computing and Artificial Intelligence, Southwest Jiaotong University Chengdu, China, in 2022 and a Master of Engineering Computer Technology degree from Southeast University Nanjing, China, in 2018. He is currently Postdoctoral Research Fellow with the Information Quantum Technology Laboratory (IQTL) at Southwest Jiaotong University Chengdu, China. He has been a reviewer for several IEEE journals and major conferences. His current research interests include Quantum Computing, Cloud Computing, Quantum algorithm, Task scheduling, Energy optimization, and Resource allocation in green computing.

Lian-Fu Wei received the Bachelor degree of Physics from Guangxi Normal University, Master degree of Pedagogy from Sichuan Normal University, and PhD degree of Engineering from Southwest Jiaotong University, China. He is a full Professor at School of Information Science and Technology, Southwest Jiaotong University, China. He has published more than 200 papers in reviewed journals, books and conference proceedings, including more than 150 papers in journals indexed in the Web of Science, Journal Citation Reports such as Information Sciences, Physics, and Electric and superconducting Electronics. Currently, he is interested on quantum computing, energy optimization and photonic communication.

Amir Rehman is a Ph.D. (Computer Science and Technology) Scholar in the School of Computing and Artificial Intelligence Chengdu, China. He has completed his Masters in Information Technology from the University of Gujrat, Lahore campus, Pakistan. He has extensive experience in the development and teaching of Computer Science to undergraduate and college students. His research mainly focuses on Machine and Deep learning, Intelligent Diagnosis, and IoMT.

Muqadar Ali received MS degree in Applied Mathematics from Nanjing University of Science and Technology (NJUST), Nanjing, China, in 2020. He is currently research students in Southwest Jiaotong University in School of Computing and Artificial Intelligence. He has been reviewer of many ICAIS 2020 conference papers. His current research includes cryptography, information security, network security and internet of things IoTs.

Syed Muhammad Waqas received the BS degree in computer science from The Islamia University of Bahawalpur Pakistan and MS degree in computer science and technology from Xi'an Jiaotong University (XJTU), China in 2015 and 2020 respectively. He is currently pursuing the Ph.D. degree in Computer Science and technology with the School of Computer Science and Technology, Xi'an Jiaotong University (XJTU), China. His research interests include computer networks, next generation networks, 5G/6G cellular networks, routing protocols and IoTs.

Fakhar Abbas received the Ph.D. degree in Information and Communication Engineering from the Key Laboratory of Information Coding and Transmission School of Southwest Jiaotong University (SWJTU), Chengdu China in 2020 and the M.S. degree in Information and Communication Engineering from Harbin Engineering University, Harbin, China, in 2015. He has been a reviewer for several IEEE journals and major conferences. His current research interests include wireless communications, vehicular ad-hoc network, 5G/6G cellular networks, C-V2X, routing protocols, Network Security, IoTs, radio resource management and algorithm designs.