Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Multi-objective workflow scheduling in heterogeneous clouds through billing-aware task-group reassignment

Shuhui Zhang[1,2]

zhangshh@sdas.org

Qilu University of Technology

Wenhao Li[1,2]

Qilu University of Technology

Lianhai Wang[1,2]

Qilu University of Technology

Shujiang Xu[1,2]

Qilu University of Technology

Wei Shao[1,2]

Qilu University of Technology

Qizheng Wang[1,2]

Qilu University of Technology

Research Article

Keywords: Heterogeneous clouds, Multi-objective workflow scheduling, Archive guidance, Billing awareness, Ant colony optimization

Posted Date: November 26th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11474769/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Multi-objective workflow scheduling in heterogeneous clouds through billing-aware task-group reassignment

Shuhui Zhang[1,2,*], Wenhao Li[1,2], Lianhai Wang[1,2], Shujiang Xu[1,2], Wei Shao[1,2], Qizheng Wang[1,2]

[1] *Key Laboratory of Computing Power Network and Information Security, Ministry of Education, Shandong Computer Science Center (National Supercomputer Center in Jinan), Qilu University of Technology (Shandong Academy of Sciences), Jinan, China*

[2] *Shandong Provincial Key Laboratory of Industrial Network and Information System Security, Shandong Fundamental Research Center for Computer Science, Jinan, China*

* Corresponding author: Shuhui Zhang; e-mail: zhangshh@sdas.org

Abstract: Task dependencies, heterogeneous resources, and discrete billing jointly complicate multi-objective workflow scheduling in clouds. This paper proposes archive-guided billing-aware ant colony optimization (AGBACO) to minimize makespan, cost, and energy consumption. The method inherits complete task-to-virtual-machine mappings from a nondominated archive and reassigns tasks sharing the same virtual machine as a group. This operation explores opportunities to release the source virtual machine and reuse the target's existing billing intervals. Adjustments involving one or two tasks further refine resource matching. Quantum-inspired genetic exploration establishes the initial archive, while conventional ant colony construction and archive-based reassignment jointly generate candidates for three-objective evaluation and archive updates. Parameter sensitivity is examined on five scientific workflows, followed by algorithm comparisons and component ablations at four task scales. Relative to five multi-objective comparison algorithms, the mean paired hypervolume gain, averaged across inputs, ranges from 8.01% to 12.24%. Solution-set quality and resource-occupancy results show that AGBACO accommodates different workflow structures and achieves favorable trade-offs between execution efficiency and resource overhead.

Keywords: Heterogeneous clouds; Multi-objective workflow scheduling; Archive guidance; Billing awareness; Ant colony optimization

1 Introduction

Cloud computing provides elastic resources for scientific computing, artificial intelligence, and large-scale data analytics. These applications commonly comprise tasks connected by data dependencies. Task ordering and resource allocation jointly affect efficiency, cost, and energy consumption, requiring coordinated scheduling of complete workflows [1–3].

Combining private and public clouds broadens resource choices while introducing trade-offs among performance, price, and power consumption. Faster virtual machines (VMs) can accelerate critical tasks. Distributing tasks creates parallelism but may increase separate VM charges and idle time. Consolidating tasks facilitates resource reuse but can prolong waiting. Scheduling must coordinate resource matching, parallelism, and occupancy overhead to obtain Pareto solutions representing different objective preferences.

Under hourly billing, shorter execution does not necessarily reduce cost. Reassigning an individual task may improve execution efficiency without reducing the source VM's billed hours. For example, shortening a billed duration from 50 to 35 minutes still incurs a one-hour charge. Scheduling therefore requires both appropriate task-to-resource matching and coordinated allocation of tasks sharing a VM. Their combined adjustment offers opportunities to balance makespan, cost, and energy consumption.

Recent studies improve workflow scheduling through heuristic initialization, heterogeneous encodings, and population-based search [4–9]. Zhang et al. coordinate global exploration and local exploitation using differentiated populations and adaptive genetic perturbations [10]. These approaches improve initial solutions and search diversity. Under hourly billing, improving an individual task assignment may leave the source VM's billed hours unchanged. Reconstructing an entire schedule may also disrupt favorable resource combinations. This study investigates how to preserve assignments from promising schedules while jointly adjusting tasks sharing a VM to improve the three-objective trade-off.

We propose archive-guided billing-aware ant colony optimization (AGBACO). Quantum-inspired genetic exploration establishes an initial archive. Conventional ant colony construction and archive-guided task-group reassignment then explore new resource

combinations and adjust resource occupancy in existing schedules. Three-objective evaluation continuously updates the archive. The main contributions are as follows.

(1) An archive-guided billing-aware task-group reassignment mechanism uses tasks sharing a source VM as the adjustment unit. It explores opportunities to release source resources and reuse target billing intervals, while small task adjustments refine heterogeneous resource matching.

(2) An ant colony search process connects complete-mapping inheritance with three-objective feedback. Quantum-inspired genetic exploration initializes the archive, and conventional construction cooperates with archive-based reassignment to develop resource configurations with different objective trade-offs.

(3) A unified model represents task dependencies, cross-cloud communication, interval-based billing, and operational energy consumption. Parameter experiments use five scientific workflows, while comparisons and component ablations cover four task scales. Solution-set quality, realized scheduling objectives, and load evolution jointly evaluate the method.

Section 2 reviews related work. Sections 3 and 4 introduce the background and system model. Sections 5 and 6 present the algorithm. Section 7 analyzes the experiments, and Section 8 concludes the paper.

2 Related work

2.1 Conventional heuristic scheduling

Heuristic methods construct schedules using task dependencies and resource performance. Kushwaha et al. combine heterogeneous earliest finish time (HEFT) initialization with Archimedes optimization [4]. Namdev and Rajak improve heuristic decisions through Q-learning [5]. Cui et al. combine HEFT, whale optimization, and Lévy flights to optimize time and cost [11]. Dependency-aware rules improve initial schedules. Local gains from individual task placements, however, do not necessarily reduce overall overhead. Further adjustment of resource combinations within complete schedules remains valuable.

2.2 Swarm intelligence optimization

Swarm intelligence methods explore resource combinations through interactions among individuals. Huang et al. design a task encoding for grey wolf optimization [6]. Gupta and Singh incorporate user-defined weights into whale optimization [7]. Abualigah et al. improve synergistic swarm optimization for task allocation [8]. Kumar and Vardhan propose IBWC, incorporating Cauchy mutation, sparrow search operators, and quasi-opposition-based learning into beluga whale optimization [12]. Customizable weights balance execution time, load, and cost. Improved encodings and update operators strengthen resource search. With hourly billing, shorter execution or a few changed assignments may still retain the original billed duration. Search must account for both matching quality and changes in VM usage.

2.3 Hybrid optimization and cooperative search

Hybrid optimization improves search through complementary strategies. Behera and Sobhanayak combine genetic algorithms and grey wolf optimization for heterogeneous cloud scheduling [9]. Gunduz et al. integrate differential evolution with cat swarm optimization [13], while Rathi et al. introduce genetic operations into a fitness-dependent optimizer [14]. Cooperative coevolutionary genetic programming jointly evolves task-ordering and resource-selection rules [15]. GMPSO combines load-aware initialization, differentiated populations, and adaptive genetic perturbations [10]. Multi-hierarchy particle swarm optimization uses workflow aggregation and resource pricing to reduce cost [16]. These strategies coordinate exploration, exploitation, and retention of promising information through operator roles. Information transfer and subsequent recombination determine whether favorable task-to-resource combinations survive. Cooperative search thus involves both strategy integration and reuse of complete configurations.

Task structure also guides cooperative search. Guo et al. form neighborhood groups from directed acyclic graph (DAG) dependencies and design group crossover and mutation for three-objective hybrid cloud scheduling [17]. Zhang et al. combine adaptive random and directed mutation for budget-constrained scheduling [18]. Shobeiri et al. integrate partial critical paths and ant colony search under deadline constraints [19]. Tao et al. propose DB-ACO with both deadline and budget constraints [20]. Dependency grouping and critical paths capture data

relationships and reduce independent treatment of coupled decisions. Dependency structure differs from resource occupancy: grouped tasks may occupy several VMs, and one VM may serve multiple branches. Structure-based groups therefore need not directly capture consolidation and billing changes. Adjustment groups can instead reflect actual occupancy in the current mapping.

2.4 Quantum-inspired and multi-objective scheduling

Quantum-inspired optimization explores resource combinations through probabilistic encoding, rotation, and observation. Hussain et al. propose the cost-aware CQGA for makespan and cost optimization in hybrid clouds [21]. Their MQGA addresses healthcare workflows with hard and soft deadlines and multiple objectives [22]. Karami et al. combine semi-greedy initialization and the nondominated sorting genetic algorithm II (NSGA-II) for fog-cloud scheduling [23]. Dynamic variable analysis guides adaptive evolutionary search by allocating effort according to variable contributions in large-scale workflows [24].

Multi-objective selection determines how promising information is retained and used. Xue et al. propose QLMOEA/D, combining enhanced Tchebycheff decomposition and Q-learning-based parameter selection to optimize workflow time, cost, and load [25]. Qiu et al. propose a dynamic multipopulation genetic algorithm (DMGA) [26]. Longest-common-subsequence information guides differentiated genetic operations that preserve favorable gene combinations and extend search coverage.

Probabilistic encoding, variable analysis, and multi-objective selection improve sampling, effort allocation, and elite retention, respectively. Independent sampling or updating of resource choices can also disrupt favorable task combinations. Maintaining a nondominated set preserves objective trade-offs, while candidate generation determines how those combinations are reused. Connecting complete-schedule retention with subsequent adjustment allows search to explore new combinations while inheriting existing configurations.

2.5 Application environments and resource constraints

Application constraints shape resource optimization. Mangalampalli et al. use deep reinforcement learning for priority-based scheduling [27]. MOMWS combines multi-objective evolution and reinforcement-based rescheduling, considering data locality, deadlines, and reuse

of rental periods [28]. Jalali Khalil Abadi et al. review deep reinforcement learning for distributed scheduling [29]. Other studies coordinate energy and performance in edge-cloud scheduling [30], select energy-efficient VM migration destinations through hyrax optimization [31], and learn multi-cloud decisions using asynchronous advantage actor–critic methods [32]. Task clustering and two-level frequency scaling also reduce energy consumption [33]. Girgis et al. propose GWTSA, partitioning workflows into branches and applying genetic search to service configurations under deadlines [34]. These studies incorporate deadlines, locality, rental periods, and power into scheduling. Resource reuse affects objectives differently: consolidation can reduce separate charges while increasing waiting or extending target VM usage. Cross-cloud adjustments also change communication overhead. Consolidation benefits require evaluation of complete execution schedules across objectives.

In multi-objective resource allocation, the strength Pareto evolutionary algorithm 2 (SPEA2) addresses IoT cloud task assignment [35]. Adaptive dynamic grouping (ADG) organizes tasks by dependencies, allocates search effort according to optimization contributions, and improves mappings through within-group task exchanges [36]. NSGA-III-GKM++ addresses cloud service selection [37]. Solution-set selection, dependency grouping, and service matching concern different decision stages. Under hourly billing, data dependencies, VM task placement, and resource prices jointly determine adjustment benefits. A consistent evaluation model helps examine their interaction.

Existing studies have improved initial solution quality, cooperative search, and resource reuse. Under hourly billing, better individual task assignments may leave VM charges unchanged, while reconstructing schedules can disrupt favorable resource configurations. AGBACO addresses these challenges through archive-guided billing-aware task-group reassignment. It retains complete mappings from promising schedules and jointly reassigns tasks sharing a VM to explore resource reuse and billing reductions within three-objective search.

3 Preliminaries

3.1 Multi-objective optimization and crowding distance

A feasible schedule x has objective vector $\mathbf{f}(x) = (T(x), C(x), E(x))$, with all three objectives minimized. Schedule x dominates y if it is no worse in every objective and strictly better in at least one. The dominance relation between x and y is

$$x \prec y \Leftrightarrow [\forall r \in \{1,2,3\},\ f_r(x) \le f_r(y)] \wedge [\exists r,\ f_r(x) < f_r(y)]. \tag{1}$$

Mutually nondominated schedules form a candidate Pareto set. Crowding distance measures local density through separations between neighboring objective vectors and favors sparsely distributed schedules during elite selection.

3.2 Quantum-inspired probabilistic encoding

The quantum-inspired state of task t_i is represented as

$$|q_i\rangle = \cos\theta_i|0\rangle + \sin\theta_i|1\rangle, \qquad \theta_i \in [0, \pi/2]. \tag{2}$$

Here, $|0\rangle$ and $|1\rangle$ represent the private and public clouds, with observation probabilities $\cos^2\theta_i$ and $\sin^2\theta_i$, respectively. After selecting a cloud type, a VM is chosen within that cloud to obtain an evaluable resource mapping.

3.3 Scheduling problem

A workflow is a DAG $G = (\mathcal{T}, \mathcal{D})$, with task set $\mathcal{T} = \{t_1, \dots, t_n\}$ and dependency edge set $\mathcal{D}$. Resource set $\mathcal{V} = \{v_1, \dots, v_m\}$ contains private and public cloud VMs. A complete schedule $x = (\boldsymbol{\pi}, \mathbf{a})$ comprises a precedence-feasible sequence and a mapping vector. Assignment $a_i = k$ places task t_i on VM v_k. Scheduling seeks three-objective nondominated solutions subject to precedence and resource non-overlap constraints.

4 System model

4.1 System architecture

The system comprises workflows, heterogeneous resource pools, and a scheduling optimizer, as shown in Fig. 1. Task workloads, dependency data, resource performance, and prices form the optimization inputs. Three-objective evaluation feeds back to the nondominated

archive, which provides schedules with different trade-offs. Table 1 summarizes the main notation.

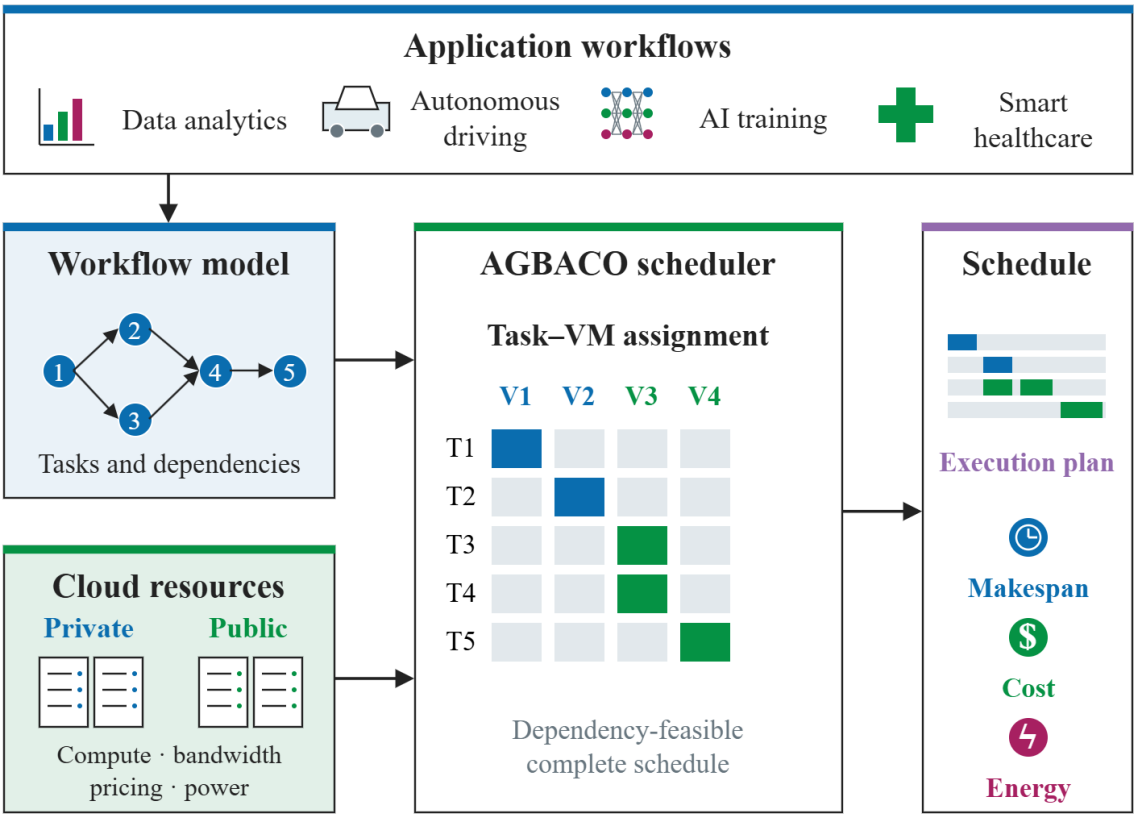

Fig. 1 Architecture for multi-objective workflow scheduling in heterogeneous clouds

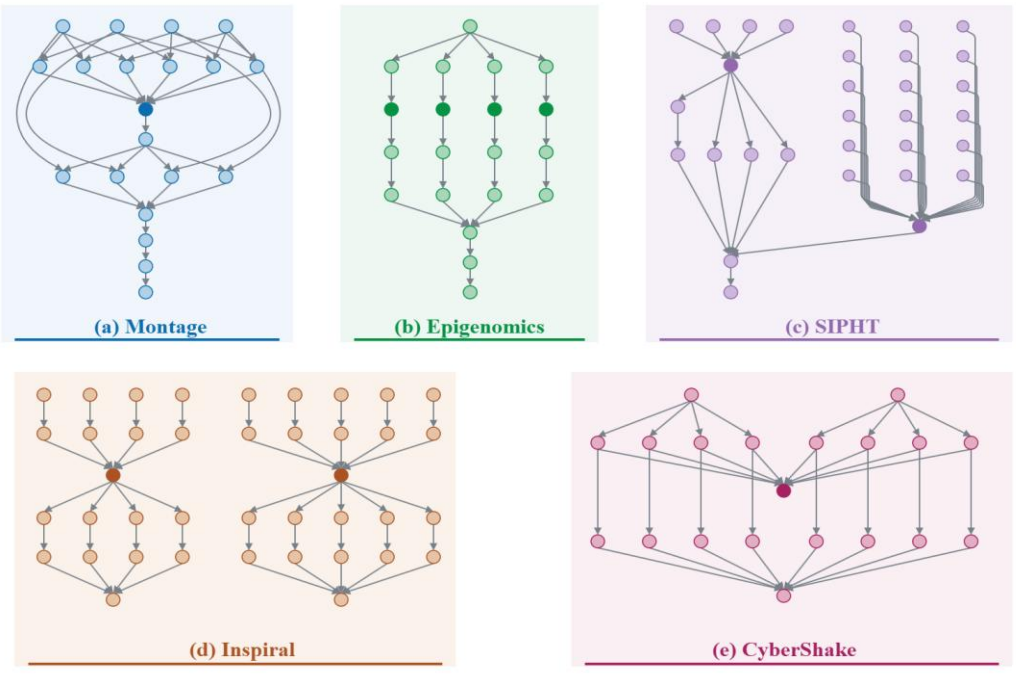

Fig. 2 Task dependencies in five representative scientific workflows

Figure 2 illustrates representative topologies of the five workflow families. Montage assembles astronomical images through projection, overlap analysis, and aggregation. Epigenomics organizes sequence processing into multistage branches. SIPHT combines parallel analysis and result integration, while Inspiral comprises waveform matching and aggregation tasks. CyberShake contains multilevel seismic computation dependencies. Parallel branches create allocation choices, while joins constrain successor start times. Together, they influence task placement, heterogeneous matching, and VM operating spans.

Table 1 Main mathematical notation

Symbol	Definition	Symbol	Definition
$G = (\mathcal{T}, \mathcal{D})$	Workflow DAG	n, m	Task and VM counts
L_i, d_{ij}	Task workload and dependency data	s_k, b_{kl}	VM speed and link bandwidth
$a_i, \boldsymbol{\pi}$	Task mapping and execution sequence	S_i, F_i	Task start and finish times
A_k, H_k, W_k	First start, last finish, and busy duration	p_k, h	VM price and billing interval
$P_k(t), \zeta_k(t)$	Instantaneous power and busy indicator	$P_k^{\text{idle}}, P_k^{\text{full}}$	Idle and full-load powers
q_C, q_E	Cross-cloud cost and energy coefficients	I_Q, I_A	Iterations in the two search stages
T, C, E	Makespan, cost, and energy consumption	T_0, C_0, E_0	Fixed normalization references
$\mathcal{A}, H$	Nondominated archive and capacity	N, K	Population size and ant count
θ_i, σ	Angle and perturbation standard deviation	τ_{ik}, η_{ik}	Pheromone and heuristic information
α, β, ρ	Pheromone and heuristic exponents; retention coefficient	$\mathcal{G}_s$	Source VM task group

4.2 Workflow execution and communication

Task t_i requires L_i million instructions (MI), and VM v_k has processing speed s_k in million instructions per second (MIPS). Its execution time is

$$e_{ik} = \frac{L_i}{s_k}. \tag{3}$$

Dependency edge (t_i, t_j) carries d_{ij} Mbit of data. Data are directly available when both tasks share a VM. Otherwise, communication time depends on the effective link bandwidth b_{kl}:

$$c_{ij} = \begin{cases} 0, & a_i = a_j, \\ d_{ij}/b_{a_i a_j}, & a_i \neq a_j. \end{cases} \tag{4}$$

Each VM executes at most one task at a time, and tasks execute non-preemptively in scheduling order. Let R_k be the availability time of VM v_k after its assigned tasks, and let $\mathrm{pred}(i)$ contain the immediate predecessors of task t_i. Start and finish times satisfy

$$S_i = \max\left\{R_{a_i}, \max_{j \in \mathrm{pred}(i)} (F_j + c_{ji})\right\}, \qquad F_i = S_i + e_{i,a_i}. \tag{5}$$

The predecessor term is zero for entry tasks. Updating VM availability sequentially ensures precedence feasibility and non-overlapping execution. Workflow makespan is

$$T(x) = \max_{1 \le i \le n} F_i. \tag{6}$$

4.3 Resource billing

Resource cost comprises VM time charges and cross-cloud data transfer charges. Let h be the billing interval and p_k the price of VM v_k per interval. A used VM is billed from the time origin to its last task completion time H_k. An unused VM incurs no time charge. With $\mathcal{V}_x$ denoting the VMs used by schedule x,

$$C_{\mathrm{VM}}(x) = \sum_{v_k \in \mathcal{V}_x} p_k \left\lceil \frac{H_k}{h} \right\rceil. \tag{7}$$

Indicator $\chi_k \in \{0,1\}$ identifies whether a VM belongs to the private or public cloud. Cross-cloud dependency edges form the set $\mathcal{D}_\times(x) = \{(t_i, t_j) \in \mathcal{D} : \chi_{a_i} \neq \chi_{a_j}\}$. Using decimal gigabytes, the transferred data volume and total cost are

$$D_\times(x) = \sum_{(t_i,t_j) \in \mathcal{D}_\times(x)} \frac{d_{ij}}{8000}, \qquad C(x) = C_{\mathrm{VM}}(x) + q_C D_\times(x). \tag{8}$$

Stepwise billing links cost to task mappings and VM usage. Shortening billed occupancy or emptying a used VM can reduce charges. Cross-cloud communication affects successor readiness, while its charges and energy consumption enter their respective objectives. The model jointly evaluates resource selection and task placement.

4.4 Energy consumption

Energy consumption comprises VM operation and cross-cloud data transfer. For each used VM v_k, let A_k be its first task start, H_k its last completion, and $W_k = \sum_{i:a_i=k} e_{ik}$ its busy duration. Its idle and full-load powers during operation are P_k^{idle} and P_k^{full}. Indicator $\zeta_k(t)$ equals 1 when the VM executes a task at time t, and 0 while waiting. Instantaneous power within its operating span is

$$P_k(t) = P_k^{\text{idle}} + \left(P_k^{\text{full}} - P_k^{\text{idle}}\right)\zeta_k(t), \qquad t \in [A_k, H_k]. \tag{9}$$

Integrating power over time gives the VM operational energy consumption:

$$E_{\text{VM}}(x) = \frac{1}{3.6\times10^6}\sum_{v_k\in\mathcal{V}_x} \int_{A_k}^{H_k} P_k\,(t)\,\mathrm{d}t. \tag{10}$$

Power is measured in watts and time in seconds; the conversion factor yields kilowatt-hours. Since the combined length of busy intervals is W_k, Eq. (10) becomes

$$E_{\text{VM}}(x) = \frac{1}{3.6\times10^6}\sum_{v_k\in\mathcal{V}_x} \left[P_k^{\text{idle}}(H_k - A_k - W_k) + P_k^{\text{full}} W_k\right]. \tag{11}$$

The power integral includes energy consumed during task execution and intervening waits. Task placement and resource selection affect busy durations and operating spans, connecting dependency structure with energy evaluation. Including cross-cloud transfer gives

$$E(x) = E_{\text{VM}}(x) + q_E D_{\times}(x). \tag{12}$$

4.5 Optimization problem

The workflow scheduling problem is formulated as

$$\min_{x\in\Omega} \left(T(x), C(x), E(x)\right), \tag{13}$$

The feasible set Ω enforces unique task assignments, precedence constraints, and non-overlapping VM execution, with $a_i \in \{1, \ldots, m\}$. All three objectives are computed from the same execution schedule, and solutions are retained in a nondominated archive.

5 Algorithm design

Figure 3 presents the two-stage AGBACO search. Quantum-inspired genetic exploration generates diverse complete mappings for the initial archive. The ant colony stage combines

conventional construction with archive-guided billing-aware task-group reassignment to explore new combinations and neighborhoods of existing schedules. Both candidate types update the archive through three-objective evaluation.

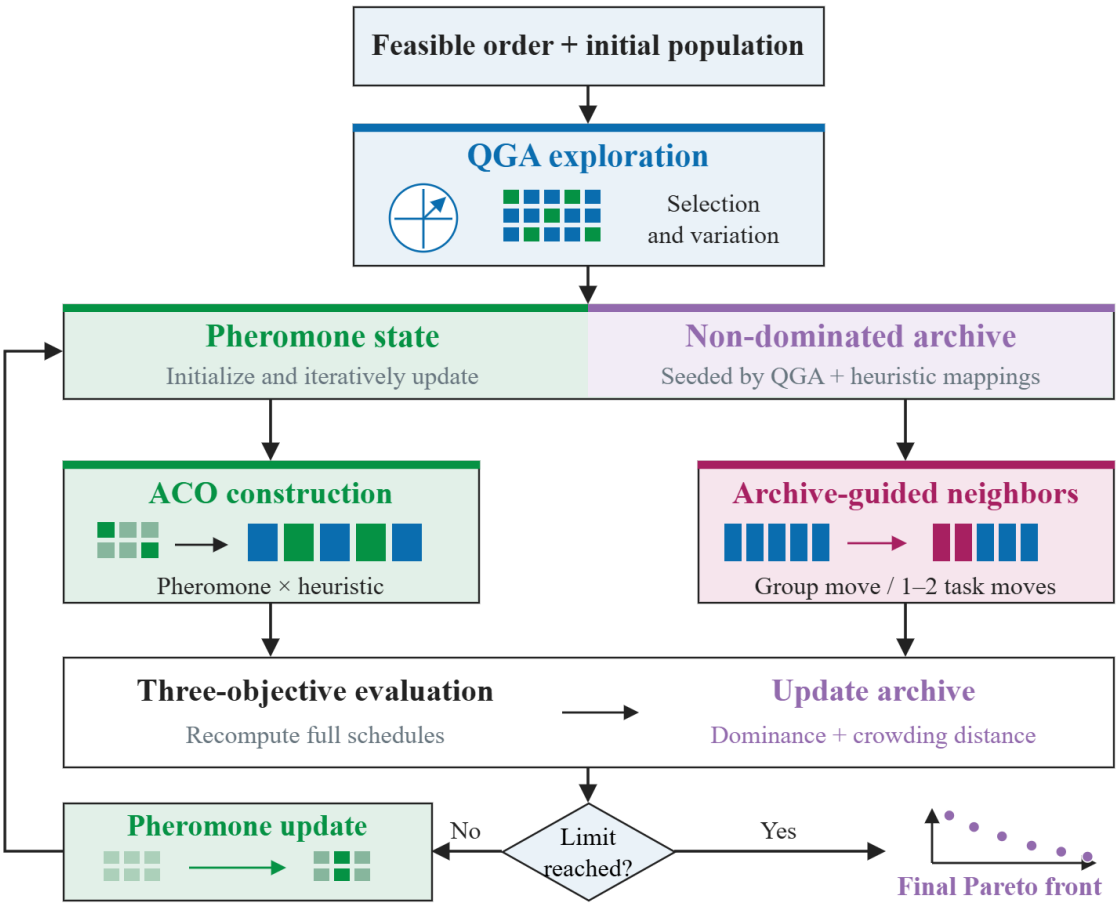

Fig. 3 Search process of AGBACO

5.1 Initial quantum-inspired genetic exploration

5.1.1 Task priorities and individual encoding

Tasks are prioritized using HEFT upward ranks [4]. With average execution time $\bar{e}_i$ and average communication time $\bar{c}_{ij}$, task priority is

$$u_i = \bar{e}_i + \max_{j \in \mathrm{succ}(i)} (\bar{c}_{ij} + u_j). \tag{14}$$

The successor term is zero for exit tasks. Among tasks whose predecessors have been scheduled, the algorithm selects the largest u_i to form precedence-feasible sequence $\boldsymbol{\pi}$. Each individual is encoded as $z = (\boldsymbol{\pi}, \mathbf{a}, \boldsymbol{\theta})$. Its resource mapping and quantum-inspired angles

describe cloud and VM choices. Initialization combines heuristic mappings and probabilistic observation, incorporating structural information and configuration diversity.

5.1.2 Selection, crossover, and quantum-inspired updates

Each iteration decodes observed complete mappings and evaluates makespan, cost, and energy consumption. Pareto dominance assigns nondomination ranks, with better ranks favored for reproduction. Crowding distance distinguishes individuals of the same rank by their objective-space distribution. For objective r, let z^+, z^- denote adjacent individuals after sorting. The crowding distance of an interior individual is

$$\mathrm{CD}(z) = \sum_{r=1}^{3} \frac{f_r(z^+) - f_r(z^-)}{f_r^{\max} - f_r^{\min}}. \tag{15}$$

Equation (15) sums normalized neighbor separations across objectives. Individuals with larger crowding distances are favored to maintain front coverage. A zero objective range contributes zero, and boundary individuals receive priority. Binary tournament selection first compares nondomination ranks and then crowding distances, balancing quality and distribution.

Uniform crossover recombines assignments task by task, randomly inheriting VM mappings, cloud types, and angle information from two parents. The precedence-feasible task sequence remains unchanged. Offspring execution schedules are recomputed in that sequence to explore combinations of parental resource choices.

Angle updates depend on nondomination rank and crowding distance. Sparsely distributed individuals in better ranks retain their angles, while others undergo random angle resets to increase exploration. Mutation adds Gaussian perturbations and clips the updated angles to the valid interval:

$$\theta_i' = \mathrm{clip}_{[0,\pi/2]}(\theta_i + \varepsilon_i), \qquad \varepsilon_i \sim \mathcal{N}(0, \sigma^2). \tag{16}$$

Here, σ controls perturbation magnitude. After updating angles, observation selects the two cloud types with probabilities $\cos^2\theta_i$ and $\sin^2\theta_i$. Sampling a VM within the selected cloud produces a new complete mapping for reevaluation. The two sampling levels extend the resource combinations explored.

The quantum-inspired genetic stage preserves promising mappings through elitism and maintains diversity through uniform crossover and probabilistic observation. Hypervolume

(HV) guides retention of a promising nondominated front. This front initializes the ant colony archive and supplies complete parent schedules for task-group reassignment.

5.2 Archive-guided billing-aware ant colony search

5.2.1 Nondominated archive and stage transition

Archive $\mathcal{A}$ stores current nondominated complete mappings. A candidate enters if no archive member dominates it, and members dominated by the candidate are removed. Duplicate objective vectors are eliminated. If capacity H is exceeded, crowding distance removes members from dense regions. Promising genetic-stage mappings and heuristic schedules initialize the archive, which subsequent candidates continuously update.

5.2.2 Ant colony construction and pheromone updates

Conventional ant colony construction selects a resource for each task in sequence $\boldsymbol{\pi}$. The probability of assigning task t_i to VM v_k is

$$P_{ik} = \frac{\tau_{ik}^{\alpha}\eta_{ik}^{\beta}}{\sum_{v_l \in \mathcal{V}} \tau_{il}^{\alpha}\eta_{il}^{\beta}}. \tag{17}$$

Here, τ_{ik} denotes pheromone and η_{ik} the inverse local composite cost. This cost combines estimated finish time, incremental charges, and task-execution and cross-cloud transfer energy using weighted terms. Objectives are normalized by fixed instance-specific references (Section 7.2), with an additional load-balancing term. The latter uses the coefficient of variation of all VM completion times after a tentative assignment, divided by one plus that coefficient. Parameters α and β control the influence of pheromone and heuristic information, respectively. Construction selects the highest-preference resource with probability 0.95; otherwise, it samples using Eq. (17). Pheromone is initialized to 10 and updated each iteration as

$$\tau_{ik}^{\text{new}} = \text{clip}_{[\tau_{\min}, \tau_{\max}]}(\max\{\tau_{\min}, \rho\tau_{ik}\} + \Delta\tau_{ik}). \tag{18}$$

Parameter ρ is the retention coefficient, and pheromone bounds are 1 and 100. The reinforcement $\Delta\tau_{ik}$ sums fitness values from promising candidates containing the corresponding task-to-resource assignment. Fitness is the inverse weighted sum of normalized makespan, cost, and energy consumption. Candidates are screened against the mean fitness of

the quantum-inspired genetic front. If none reaches this threshold, all valid candidates contribute.

5.2.3 Archive-guided billing-aware task-group reassignment

Billing-aware task-group reassignment uses a complete archived schedule as its parent and jointly adjusts tasks sharing a VM (Fig. 4). For source VM v_s, the task group is $\mathcal{G}_s(x) = \{t_i \in \mathcal{T}: a_i = s\}$. After selecting another used VM v_d, all group members are remapped as

$$a_i' = \begin{cases} d, & t_i \in \mathcal{G}_s(x), \\ a_i, & t_i \notin \mathcal{G}_s(x). \end{cases} \tag{19}$$

The candidate is rescheduled in precedence-feasible order. The source VM becomes unused, and the target executes its tasks. The last completion time of an unused VM is defined as zero. The complete cost change is

$$\Delta C = C(x') - C(x) = \sum_{k=1}^{m} p_k \left(\left\lceil \frac{H_k'}{h} \right\rceil - \left\lceil \frac{H_k}{h} \right\rceil \right) + q_C [D_\times(x') - D_\times(x)]. \tag{20}$$

Here, H_k' is the last completion time of VM v_k after rescheduling. Emptying the source removes its time charges, while charges for the target and other affected resources are recomputed. Reassignment reduces total cost when source savings exceed additional occupancy and communication charges. If added tasks finish within the target's existing billing intervals, they reuse time already paid for.

In Fig. 4, two equally priced VMs initially incur one billed hour each. If the target completes the reassigned group within its original hour, the total charge falls from two VM-hours to one. The example illustrates how coordinated reassignment can eliminate separately billed occupancy.

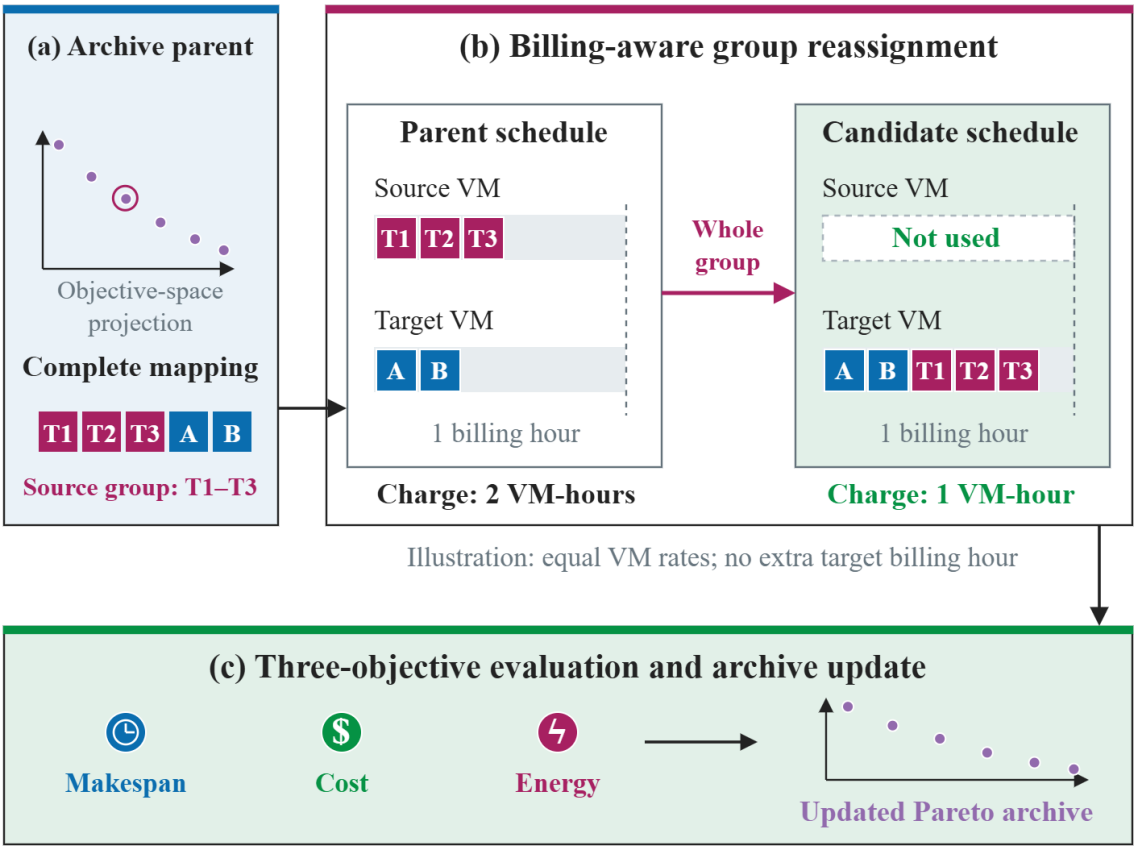

Fig. 4 Archive-guided billing-aware task-group reassignment

Approximately one-fifth of ant colony candidates use conventional construction. The remainder copy a complete parent mapping selected uniformly from the archive. Approximately one-third of archive-neighborhood candidates use whole-group reassignment; the others adjust one or two tasks to refine resource matching. Whole-group moves require at least two used VMs and randomly select distinct source and target VMs. Each candidate undergoes independent three-objective evaluation and nondominated comparison. The updated archive supports subsequent search.

5.2.4 Final schedule selection

The final archive provides nondominated schedules with different objective trade-offs. For comparisons of realized execution objectives, the three objectives undergo min–max normalization within each run's archive, denoted by $\tilde{f}_r(x)$. The representative schedule minimizes

$$R(x) = \sqrt{\sum_{r=1}^{3} \tilde{f}_r\,(x)^2} + \lambda_R(1 - J(x)). \tag{21}$$

A zero objective range yields a normalized value of zero. Term $J(x)$ is Jain's index of VM busy durations, with $\lambda_R = 0.5$. Unused VMs have zero busy duration, giving

$$J(x) = \frac{\left(\sum_{k=1}^{m} W_k\right)^2}{m \sum_{k=1}^{m} W_k^2}. \tag{22}$$

The complete archive is evaluated using HV and inverted generational distance (IGD). The selected schedule supplies the three realized execution objectives.

6 Algorithm procedure

6.1 Search procedure

Algorithm 1 gives dependency-aware task ordering. Algorithm 2 summarizes initial exploration, archive-guided search, and final schedule selection.

Algorithm 1 Dependency-aware task ordering	
Input	Workflow $G = (\mathcal{T}, \mathcal{D})$, execution costs e_{ik}, mean communication costs $\bar{c}_{ij}$
Output	Precedence-feasible task sequence $\boldsymbol{\pi}$
1	Compute u_i for all tasks in reverse topological order using the upward-rank recurrence.
2	Initialize the ready set with entry tasks and set $\boldsymbol{\pi}$ to empty.
3	while the ready set is nonempty do
4	Select the task with largest u_i, append t_i to $\boldsymbol{\pi}$, and remove it from the ready set.
5	Update unscheduled-predecessor counts; add successors whose predecessors have all been scheduled.
6	end while; return $\boldsymbol{\pi}$.

Algorithm 2 Main procedure of AGBACO	
Input	G, resources $\mathcal{V}$, population size N, ant count K, archive capacity H, stage termination criteria
Output	Nondominated archive $\mathcal{A}$ and representative schedule x^*
1	Generate task sequence $\boldsymbol{\pi}$ using Algorithm 1; initialize the quantum-inspired population and evaluate complete mappings.
2	while the quantum-inspired genetic termination criterion is unmet do
3	Compute nondomination ranks and crowding distances; retain a promising front using HV.
4	Update angles; apply selection, uniform crossover, and mutation; evaluate offspring and retain elites.
5	end while
6	Initialize $\mathcal{A}$ with the genetic-stage front and heuristic mappings; initialize the pheromone matrix.
7	while the ant colony termination criterion is unmet do
8	Initialize the candidate set for this iteration.
9	for up to K candidates, while the termination criterion remains unmet do
10	if conventional construction is selected then
11	Select VMs for all tasks using pheromone and heuristic information.
12	else
13	Select a parent uniformly from $\mathcal{A}$ and copy its complete mapping.
14	Reassign a source VM's entire task group, or adjust one or two tasks.
15	end if
16	Reschedule in precedence-feasible order; evaluate T, C, E and store the candidate.
17	end for
18	Merge candidates and archive; retain nondominated members and truncate to capacity H using crowding distance.
19	Apply pheromone evaporation and reinforcement from promising schedules.
20	end while; select x^* using the compromise criterion and return $\mathcal{A}$.

6.2 Complexity analysis

AGBACO's main computational costs arise from population search, candidate evaluation, and archive updates. Let the quantum-inspired genetic and ant colony stages run for I_Q and I_A iterations, respectively. Initialization computes task priorities and execution costs in $O(nm + m^2 + |\mathcal{D}| + n \log n)$ time. Three-objective evaluation of one complete schedule requires $O(n + |\mathcal{D}| + m)$ time.

Individual updates and evaluation in each genetic-stage iteration require $O(N(n + |\mathcal{D}| + m))$ time, with a conservative front-selection bound of $O(N^{7/2})$. Each ant colony iteration requires $O(Km(n + |\mathcal{D}| + m))$ time for candidate generation and evaluation, and $O((H + K)^2)$ for archive updates. With fixed VM count, population size, ant count, and archive capacity, the total time complexity, including initial sorting, simplifies to

$$O(n \log n + (I_Q + I_A)(n + |\mathcal{D}|)). \tag{23}$$

The task graph requires $O(n + |\mathcal{D}|)$ storage. With fixed resource and population sizes and archive capacity, task mappings, pheromone matrices, and candidate sets each require $O(n)$ storage. The total space complexity is

$$O(n + |\mathcal{D}|). \tag{24}$$

7 Experimental results and analysis

7.1 Experimental environment and settings

Experiments used WorkflowSim, built on CloudSim [10,36], to record makespan, cost, energy consumption, and load. Simulations ran on an Intel Xeon Platinum 8347C processor with 240 GB of memory. The heterogeneous resource pool contained eight private cloud VMs and twelve public cloud VMs, configured as shown in Table 2.

Public cloud prices follow the publicly available Amazon Web Services (AWS) EC2 on-demand pricing framework [38]. Fixed representative parameters distinguish general-purpose, compute-oriented, and memory-oriented resources; private cloud prices reflect internal amortized costs. The experiments use discrete hourly billing. All methods share the resources, prices, and charging rules in Table 2 to compare resource selection and billed occupancy across task combinations.

Each VM has one processing element. The billing interval is $h = 3600$ s, the cross-cloud transfer price is $q_C = 0.02$ USD/GB, and its energy coefficient is $q_E = 0.01$ kWh/GB. All algorithms share execution, communication, cost, and energy models, so objective differences arise from the schedules themselves.

Table 2 Heterogeneous VM configurations

Cloud	Resource type	Count	Speed (MIPS)	Memory (GB)	Bandwidth (Mbit/s)	Price (USD/h)	Idle/full-load power (W)
Private	General	4	1000	2	800	0.0400	80/160
Private	Compute	3	1400	4	1000	0.0480	65/190
Private	Memory	1	1200	8	900	0.0550	75/200
Public	General	6	1200	2	1000	0.1008	90/180
Public	Compute	4	1800	4	1200	0.0890	70/210
Public	Memory	2	1500	8	1100	0.1260	85/220

Five scientific workflows, CyberShake, Epigenomics, Inspiral, Montage, and SIPHT, were configured at task scales of 100, 400, 700, and 1000. These combinations provide 20 inputs.

The experiments examine parameter sensitivity, algorithm comparisons, and component ablations. Parameter analysis uses ten independent seeds at the largest configured scale. Comparisons and ablations cover all 20 inputs with 30 independent seeds each. Every stochastic search algorithm has a common limit of 20,000 complete-schedule evaluations, including initialization and new candidates. AGBACO allocates approximately half to each stage. HEFT serves as a deterministic scheduling reference.

Comparison methods include NSGA-II, MOEA/D [25], ADG [36], SPEA2 [35], DMGA [26], and HEFT. They cover nondominated sorting, decomposition, dynamic grouping, strength-based archives, multipopulation evolution, and dependency-aware heuristic scheduling. DMGA uses a three-objective adaptation. Component ablations use conventional quantum-inspired genetic search (QGA), ant colony optimization (ACO), and genetic–ant colony optimization (GAACO). These controls evaluate how probabilistic genetic exploration, pheromone-based construction, and archive-guided billing-aware task-group reassignment cooperate within the complete framework.

NSGA-II, ADG, SPEA2, and DMGA use populations of 100 and crossover probability 0.9. ADG's evolutionary backbone uses a reference-direction division parameter of 12. SPEA2 has archive capacity 100. MOEA/D uses 105 decomposition subproblems and a neighborhood

of 20. All methods use precedence-feasible discrete mappings and decoding with the same three-objective evaluation model.

AGBACO's defaults are population size $N = 100$, angle perturbation standard deviation $\sigma = 0.10\pi$, ant count $K = 100$, retention coefficient $\rho = 0.6$, pheromone exponent $\alpha = 1$, and heuristic exponent $\beta = 2$. Archive capacity is $H = 100$, and representative-schedule selection uses $\lambda_R = 0.5$. Stochastic algorithms use paired seeds on each input. A common configuration is retained across workflows and scales in the formal comparisons.

7.2 Evaluation metrics and statistical methods

Solution-set quality is measured by HV and IGD. For each workflow input, objectives are normalized using fixed makespan, cost, and energy references T_0, C_0, E_0:

$$\hat{\mathbf{f}}(x) = \left(\frac{T(x)}{T_0}, \frac{C(x)}{C_0}, \frac{E(x)}{E_0}\right). \tag{25}$$

Reference T_0 sums task durations on the slowest VM and communication durations for all dependency edges at the lowest link bandwidth. Reference C_0 sums the largest VM charge for each separately billed task and cross-cloud charges for all dependency data. Reference E_0 combines full-load energy of all VMs over T_0 with transfer energy for all dependency data. These references are fixed before search and shared by all methods and repetitions on the same input. HV uses reference point $\mathbf{r} = (1.1,1.1,1.1)$ and measures the objective-space volume covered by the archive:

$$\mathrm{HV}(\mathcal{A}) = \mathrm{vol}\left(\bigcup_{x\in\mathcal{A}}([\hat{f}_1(x), r_1] \times [\hat{f}_2(x), r_2] \times [\hat{f}_3(x), r_3])\right). \tag{26}$$

If an objective exceeds the corresponding reference coordinate, its interval is empty and contributes zero volume. Larger HV indicates better combined convergence and coverage. For each input, normalized final archives from all runs and methods are pooled with the HEFT solution. Their nondominated union forms a common empirical reference front $\mathcal{P}^*$. IGD is defined as

$$\mathrm{IGD}(\mathcal{A}, \mathcal{P}^*) = \frac{1}{|\mathcal{P}^*|}\sum_{\mathbf{y}\in\mathcal{P}^*} \min_{x\in\mathcal{A}} \| \mathbf{y} - \hat{\mathbf{f}}(x) \|_2. \tag{27}$$

Smaller IGD indicates closer approximation and broader coverage of the common empirical front. Realized scheduling objectives are arithmetic means of the representative

schedules across runs, with all three values taken from the same schedule. Error bars for solution-set metrics and realized objective means show bootstrap 95% confidence intervals.

HV is compared using paired stochastic runs. For the same input and seed, let $\mathrm{HV}_b^{\mathrm{AGB}}$ and $\mathrm{HV}_b^{\mathrm{cmp}}$ denote the HV values of AGBACO and a comparison algorithm, respectively. The mean paired relative gain is

$$\overline{\Delta} = \frac{100\%}{N_{\mathrm{rep}}} \sum_{b=1}^{N_{\mathrm{rep}}} \frac{\mathrm{HV}_b^{\mathrm{AGB}} - \mathrm{HV}_b^{\mathrm{cmp}}}{\mathrm{HV}_b^{\mathrm{cmp}}}, \tag{28}$$

Here, $N_{\mathrm{rep}} = 30$. Paired HV differences are tested using two-sided Wilcoxon signed-rank tests, with Holm correction for multiple comparisons within each family. Paired resampling provides 95% confidence intervals. Cross-input summaries equally average paired relative gains across the 20 inputs, avoiding dependence on absolute objective scales of individual workflows.

7.3 Parameter sensitivity

One-factor-at-a-time experiments examine population size, angle perturbation magnitude, ant count, pheromone retention, and pheromone exponent. Table 3 lists their values. One parameter changes at a time while the others retain defaults, producing 16 distinct configurations after deduplication.

Table 3 Parameter ranges and default settings

Parameter	Definition	Values	Default
N	Population size	50, 100, 150, 200	100
σ/π	Normalized angle perturbation	0.05, 0.10, 0.15, 0.20	0.10
K	Ant count	25, 50, 100, 200	100
ρ	Pheromone retention	0.4, 0.5, 0.6, 0.8	0.6
α	Pheromone exponent	0.5, 1.0, 1.5, 2.0	1.0

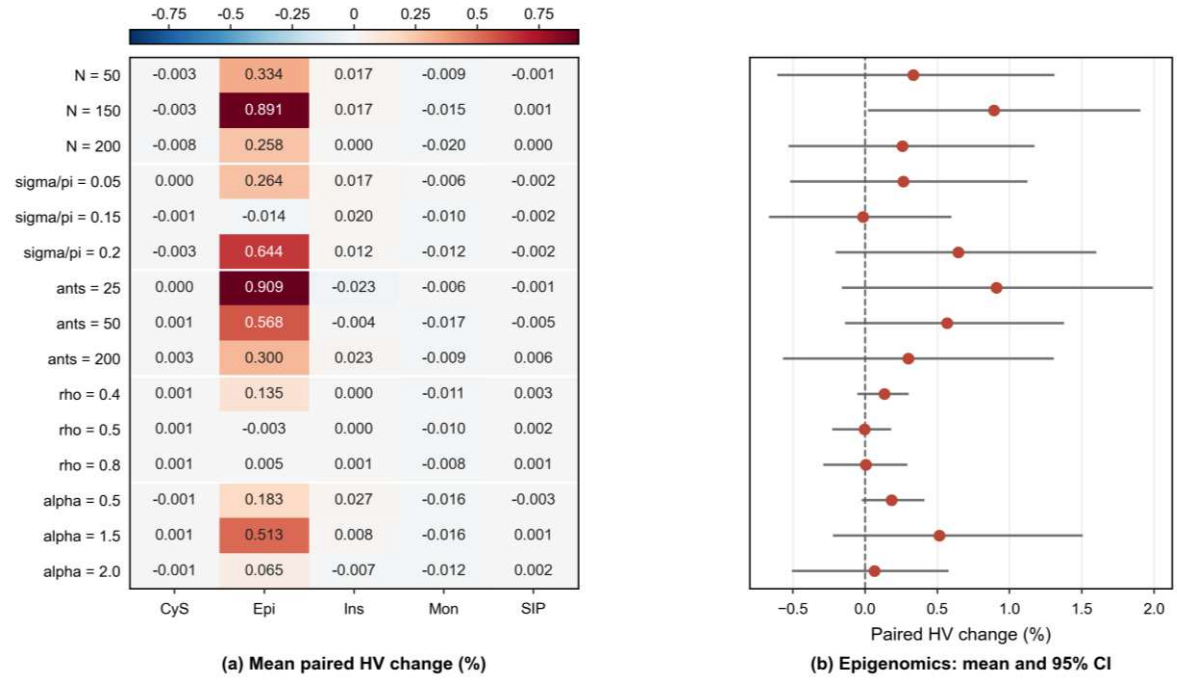

Fig. 5 Three-objective solution-set quality under different parameter settings

Figure 5 shows mean paired HV changes relative to the default configuration. Changes remain below 0.03% for CyberShake, Inspiral, Montage, and SIPHT, indicating low sensitivity within the tested ranges. For Epigenomics, an ant count of 25 yields an approximately 0.91% mean gain. Under the same search limit, smaller candidate batches increase update frequency and may allow earlier use of revised pheromone and archive information.

Population size affects initial configuration coverage, while candidate batch size determines the pace of search and updates. Considering all five workflows, subsequent experiments use the common defaults in Table 3 to assess cooperation between archive-guided search and conventional construction.

7.4 Multi-objective solution-set quality

Figure 6 compares HV and empirical IGD across five workflows and four task scales. Table 4 summarizes quality at the largest scale and mean search time across inputs. AGBACO achieves the highest mean HV on all 20 inputs and the lowest mean empirical IGD on 19. Relative to the five multi-objective comparison algorithms, its cross-input mean paired HV gains range from 8.01% to 12.24%. Paired differences on every input remain significant at the 0.05 level after Holm correction. The quality advantage thus extends across inputs and comparison methods.

Table 4 Solution-set quality at the largest task scale and mean search time across inputs

Workflow	Metric	AGBACO	NSGA-II	MOEA/D	ADG	SPEA2	DMGA	HEFT
CyberShake	HV	1.25273	1.19039	1.15852	1.18942	1.18922	1.19184	1.19101
	IGD	0.00309	0.04367	0.04008	0.04362	0.04234	0.03542	0.05856
Epigenomics	HV	0.96754	0.57848	0.44152	0.57598	0.56335	0.64441	0.88760
	IGD	0.04974	0.29903	0.41217	0.30163	0.31145	0.24818	0.10291
Inspiral	HV	1.21850	1.14776	1.12203	1.14521	1.14187	1.14917	1.19924
	IGD	0.01006	0.05124	0.05526	0.05155	0.05191	0.05064	0.05136
Montage	HV	1.22923	1.17803	1.16518	1.17840	1.17678	1.17877	1.20185
	IGD	0.00187	0.13441	0.09347	0.13448	0.13126	0.12278	0.14993
SIPHT	HV	1.22248	1.16233	1.12943	1.15613	1.15830	1.15893	1.20981
	IGD	0.00529	0.04749	0.05360	0.04863	0.04846	0.04812	0.04753
Mean search time (s)		14.09	13.30	13.12	13.38	14.27	14.86	—

The magnitude of improvement varies across workflows. At the largest scale, paired HV gains against different comparison methods range from 4.28% to 8.60% for CyberShake, Inspiral, Montage, and SIPHT. For Epigenomics, they range from 50.63% to 120.40%. The first four workflows demonstrate improvements across dependency structures. The larger Epigenomics gains are consistent with greater scope for occupancy adjustments along long execution branches.

Archive-guided search offers complementary adjustment granularities. Whole-group reassignment inherits complete mappings and explores configurations that release source VMs or reuse target billing intervals. Small task adjustments refine heterogeneous matching. Three-objective evaluation retains different trade-offs in a shared archive for subsequent search. Sections 7.5 and 7.7 examine the resulting execution objectives and resource occupancy.

In Table 4, AGBACO's mean search time is 14.09 s, compared with 13.12–14.86 s for the other multi-objective algorithms. Under a common search limit, AGBACO obtains better solution-set quality at comparable search times. Complete-mapping reuse and bounded archive search provide an efficient basis for exploring resource combinations.

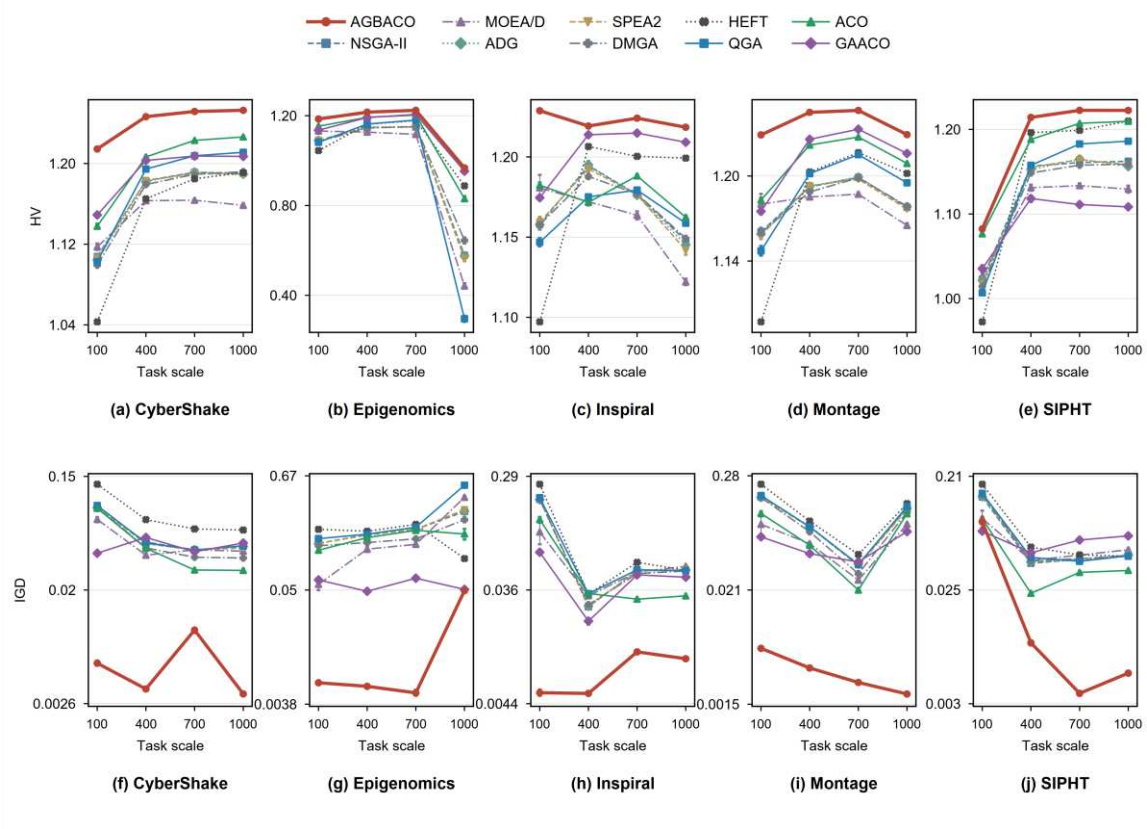

Fig. 6 HV and IGD in algorithm comparisons and component ablations; top row: HV; bottom row: IGD on logarithmic axes

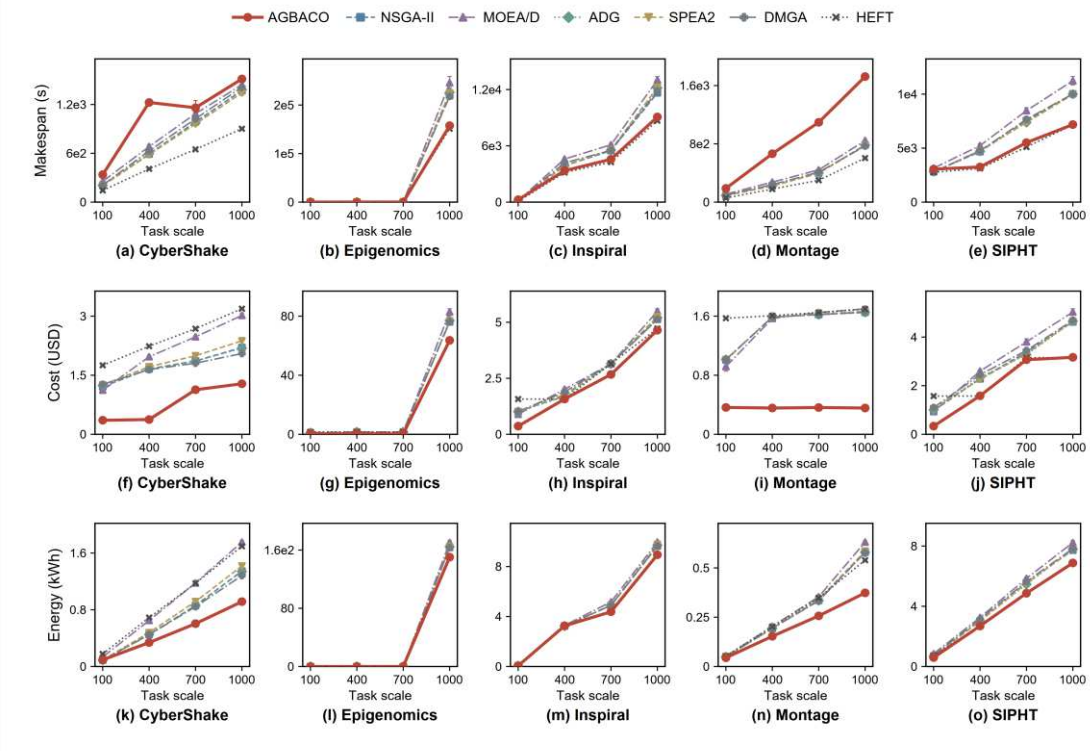

Fig. 7 Mean makespan, cost, and energy consumption across workflows and task scales

7.5 Makespan, cost, and energy consumption

Figure 7 compares mean makespan, cost, and energy consumption for seven methods at four task scales. The three objectives of each representative schedule show the resulting scheduling trade-offs.

At the largest scale, AGBACO achieves lower mean makespan, cost, and energy consumption than all five multi-objective comparison algorithms on Epigenomics, Inspiral, and SIPHT. These schedules improve execution efficiency and resource overhead together, consistent with their higher HV and lower empirical IGD.

CyberShake and Montage schedules place greater emphasis on resource economy, with clear cost and energy advantages and a makespan trade-off. Whole-group reassignment generates candidates that can reduce separately billed occupancy. The three-objective archive retains trade-offs among resource combinations, supporting exploration of such configurations. Together, Table 4 and Fig. 7 show how improved solution sets and representative-schedule preferences accommodate different workflow structures.

7.6 Component ablations

Component ablations compare AGBACO with conventional QGA, ACO, and GAACO using the same inputs and paired seeds. They assess the combined effects of initial exploration, ant colony construction, and archive exploitation. Figure 6 and Table 5 report the results.

AGBACO has higher mean HV than all three component controls on every input. Cross-input mean paired gains over QGA, ACO, and GAACO are 16.35%, 3.38%, and 3.73%, respectively. Paired differences pass significance testing after Holm correction. HV and empirical IGD in Table 5 describe objective-space coverage and approximation to the common empirical front, jointly supporting the complete framework's solution-set quality.

Table 5 Mean HV and IGD for component ablations at the largest task scale

Workflow	Metric	AGBACO	QGA	ACO	GAACO
CyberShake	HV	1.25273	1.21110	1.22635	1.20702
	IGD	0.00309	0.04412	0.02835	0.04638
Epigenomics	HV	0.96754	0.29560	0.83041	0.95272
	IGD	0.04974	0.54215	0.17948	0.05113
Inspiral	HV	1.21850	1.15868	1.16239	1.20917
	IGD	0.01006	0.05091	0.03203	0.04519
Montage	HV	1.22923	1.19521	1.20889	1.21594
	IGD	0.00187	0.14060	0.11885	0.07874
SIPHT	HV	1.22248	1.18607	1.20957	1.10843
	IGD	0.00529	0.04786	0.03635	0.06956

At the largest scale, paired HV gains over ACO are 2.15%, 4.83%, 1.68%, and 1.07% for CyberShake, Inspiral, Montage, and SIPHT, respectively. SIPHT improves by 10.29% over GAACO. Epigenomics gains 16.68% over ACO and 1.56% over GAACO. These differences support complementary roles for initial exploration and subsequent development of resource combinations. Conventional construction creates new mappings, while archive-guided billing-aware task-group reassignment inherits schedules and adjusts task placement. Three-objective feedback connects these processes, supporting exploration of both new combinations and neighborhoods of existing configurations.

7.7 Load evolution and resource configurations

Figure 8 shows mean private and public cloud load profiles at the largest scale. Time is normalized by each run's makespan; load is the mean busy fraction across all VMs in the corresponding pool. Table 6 summarizes operating spans and within-span idle times of used VMs. A span extends from first task start to last completion; idle time subtracts busy duration from that span. Each measure is summed over used VMs before averaging across runs. Comparison ranges span the means of the five multi-objective comparison algorithms.

Table 6 Summed VM operating spans and idle times at the largest task scale (10^3 s)

Workflow	AGBACO Total span	AGBACO Total idle time	Comparison methods Span range	Comparison methods Idle-time range
CyberShake	17.17	1.59	23.25 ～ 24.52	6.28 ～ 8.18
Epigenomics	2867.60	10.88	3467.23 ～ 3761.66	549.53 ～ 807.50
Inspiral	172.35	2.51	203.49 ～ 217.80	30.03 ～ 42.59
Montage	6.56	0.24	12.74 ～ 14.14	4.31 ～ 5.39
SIPHT	133.38	4.05	173.08 ～ 195.28	42.80 ～ 62.98

Epigenomics, Inspiral, and SIPHT maintain high busy fractions during their main execution phases. AGBACO's summed within-span idle times are 10.88, 2.51, and 4.05 thousand seconds, respectively, below those of all five multi-objective comparison algorithms. Together with the three-objective improvements in Fig. 7, these results indicate compact parallel execution with less resource waiting, coordinating efficiency and overhead.

CyberShake and Montage concentrate execution on public cloud resources, with low private cloud busy fractions. Their summed operating spans and idle times are below those of the five comparison algorithms, consistent with their cost and energy advantages. Pool-wide mean busy fractions include unused VMs and should be interpreted alongside operating spans of used resources. The results reveal different trade-offs from compact parallel execution and concentrated allocation, consistent with an archive retaining diverse resource combinations.

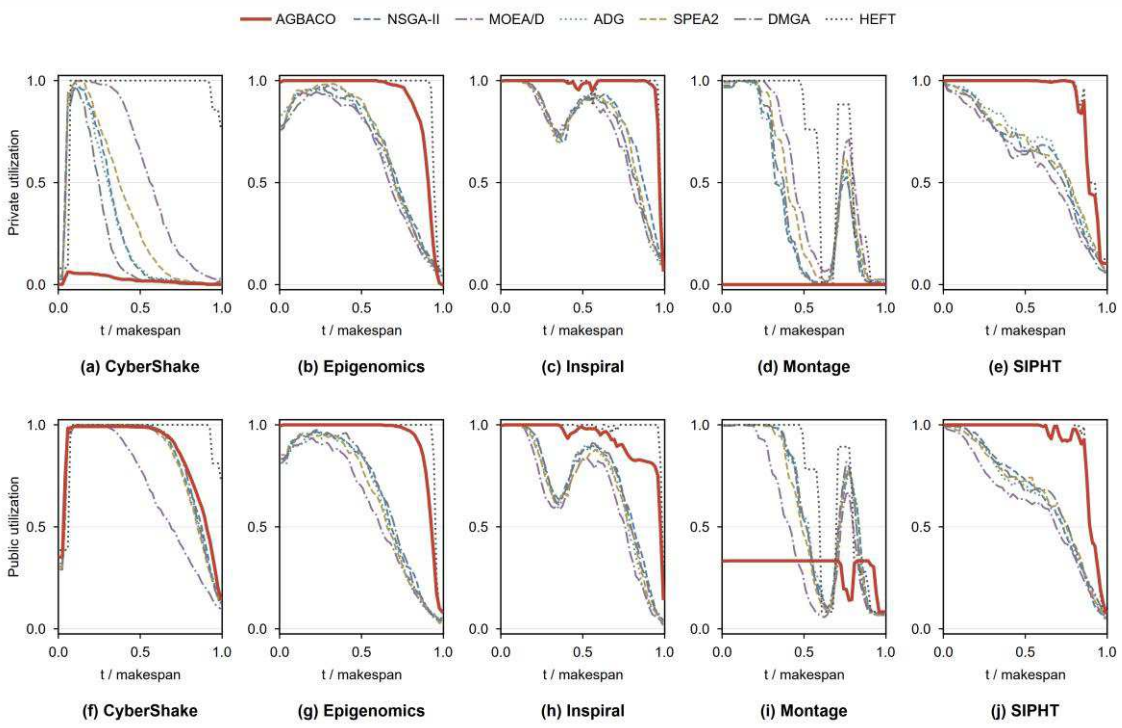

Fig. 8 Private and public cloud load profiles at the largest task scale

8 Conclusions

This paper proposes AGBACO for three-objective workflow scheduling across heterogeneous private and public clouds. It integrates complete-mapping inheritance and billing-aware task-group reassignment into ant colony search. Quantum-inspired genetic exploration establishes the initial archive. Whole-group reassignment changes VM usage, small task adjustments refine heterogeneous matching, and conventional construction expands resource combinations. Joint evaluation of makespan, cost, and energy consumption updates the archive, connecting configuration reuse with resource-occupancy adjustments.

Experiments on five workflows at four task scales show that AGBACO achieves the highest mean HV on all 20 inputs and the lowest mean empirical IGD on 19. Cross-input mean paired HV gains range from 8.01% to 12.24% over five multi-objective algorithms and from 3.38% to 16.35% over conventional QGA, ACO, and GAACO. Realized objectives and resource occupancy further demonstrate coordinated execution efficiency and resource overhead across workflow structures. Future work will examine how workflow structure, resource heterogeneity, and billing granularity influence reassignment benefits to improve

candidate selection. Archive sharing and coordinated task-group allocation will also be investigated for joint scheduling of multiple workflows, aiming to improve shared-resource utilization with modest search overhead.

Statements and Declarations

Competing interests: The authors declare that they have no known competing financial interests or personal relationships that could have influenced the work reported in this paper.

Acknowledgments

This work was supported by the National Key R&D Program of China(Grant No. 2024YFB2906502), the Taishan Scholars Program (No. tsqn202312231), the project ZR2024MF104 supported by Shandong Provincial Natural Science Foundation.

Data availability

The data supporting this study are available upon reasonable request.

References

[1] Mahdizadeh M, Montazerolghaem A, Jamshidi K (2025) Task scheduling and load balancing in SDN-based cloud computing: A review of relevant research. Journal of Engineering Research 13(4): 3132-3146. https://doi.org/10.1016/j.jer.2024.11.002

[2] Kathole A B, Vhatkar K, Lonare S, et al (2025) Optimization-based resource scheduling techniques in cloud computing environment: A review of scientific workflows and future directions. Computers and Electrical Engineering 123: 110080. https://doi.org/10.1016/j.compeleceng.2025.110080

[3] Sanjalawe Y, Al-E'mari S, Fraihat S, et al (2025) AI-driven job scheduling in cloud computing: a comprehensive review. Artificial Intelligence Review 58(7): 197. https://doi.org/10.1007/s10462-025-11208-8

[4] Kushwaha S, Singh R S, Prajapati K (2025) Multi-Objective Workflow Scheduling in Cloud Using Archimedes Optimization Algorithm. Concurrency and Computation: Practice and Experience 37(4-5): e8393. https://doi.org/10.1002/cpe.8393

[5] Namdev K, Rajak R (2025) Enhancing heuristic workflow scheduling algorithms using Q-learning technique in cloud computing. Cluster Computing 28(6): 409. https://doi.org/10.1007/s10586-025-05217-z

[6] Huang X, Xie M, An D, et al (2024) Task scheduling in cloud computing based on grey wolf optimization with a new encoding mechanism. Parallel Computing 122: 103111. https://doi.org/10.1016/j.parco.2024.103111

[7] Gupta S, Singh R S (2024) User-defined weight based multi objective task scheduling in cloud using whale optimization algorithm. Simulation Modelling Practice and Theory 133: 102915. https://doi.org/10.1016/j.simpat.2024.102915

[8] Abualigah L, Hussein A M, Almomani M H, et al (2024) Improved synergistic swarm optimization algorithm to optimize task scheduling problems in cloud computing. Sustainable Computing: Informatics and Systems 43: 101012. https://doi.org/10.1016/j.suscom.2024.101012

[9] Behera I, Sobhanayak S (2024) Task scheduling optimization in heterogeneous cloud computing environments: A hybrid GA-GWO approach. Journal of Parallel and Distributed Computing 183: 104766. https://doi.org/10.1016/j.jpdc.2023.104766

[10] Zhang P, Gao J, Tan L, et al (2025) Genetically-modified multi-population particle swarm optimization for workflow scheduling in cloud environment. Swarm and Evolutionary Computation 98: 102113. https://doi.org/10.1016/j.swevo.2025.102113

[11] Cui M, Wang Y (2025) An Effective QoS-Aware Hybrid Optimization Approach for Workflow Scheduling in Cloud Computing. Sensors 25(15): 4705. https://doi.org/10.3390/s25154705

[12] Kumar R, Vardhan M (2025) IBWC: a user-centric approach to multi-objective cloud task scheduling using improved beluga whale optimization. Knowledge and Information Systems 67(4):3423–3457. https://doi.org/10.1007/s10115-024-02325-x

[13] Gunduz A, Senan S, Gurkas-Aydin Z (2025) A hybrid DECSO algorithm for efficient multi objective task scheduling in cloud computing environments. Cluster Computing 28(13): 848. https://doi.org/10.1007/s10586-025-05586-5

[14] Rathi S, Nagpal R, Srivastava G, et al (2024) A multi-objective fitness dependent optimizer for workflow scheduling. Applied Soft Computing 152: 111247. https://doi.org/10.1016/j.asoc.2024.111247

[15] Zaki T S, Zeiträg Y, Neves R, et al (2024) A cooperative coevolutionary genetic programming hyper-heuristic for multi-objective makespan and cost optimization in cloud workflow scheduling. Computers & Operations Research 172: 106805. https://doi.org/10.1016/j.cor.2024.106805

[16] Lu C, Zhu J, Huang H, et al (2024) A multi-hierarchy particle swarm optimization-based algorithm for cloud workflow scheduling. Future Generation Computer Systems 153: 125-138. https://doi.org/10.1016/j.future.2023.11.030

[17] Guo Y, Liu B, Lin W, et al (2025) Dynamic neighborhood grouping-based multi-objective scheduling algorithm for workflow in hybrid cloud. Future Generation Computer Systems 166: 107633. https://doi.org/10.1016/j.future.2024.107633

[18] Zhang Y, Zhang L, Cao B, et al (2026) Adaptive-oriented mutation snake optimizer for scheduling budget-constrained workflows in heterogeneous cloud environments. Future Generation Computer Systems 175: 108118. https://doi.org/10.1016/j.future.2025.108118

[19] Shobeiri P, Akbarian Rastaghi M, Abrishami S, et al (2024) PCP–ACO: a hybrid deadline-constrained workflow scheduling algorithm for cloud environment. The Journal of Supercomputing 80(6): 7750-7780. https://doi.org/10.1007/s11227-023-05753-8

[20] Tao S, Xia Y, Ye L, et al (2024) DB-ACO: A Deadline-Budget Constrained Ant Colony Optimization for Workflow Scheduling in Clouds. IEEE Transactions on Automation Science and Engineering 21(2): 1564-1579. https://doi.org/10.1109/tase.2023.3247973

[21] Hussain M, Wei L F, Rehman A, et al (2024) Cost-aware quantum-inspired genetic algorithm for workflow scheduling in hybrid clouds. Journal of Parallel and Distributed Computing 191: 104920. https://doi.org/10.1016/j.jpdc.2024.104920

[22] Hussain M, Wei L F, Abbas F, et al (2022) A multi-objective quantum-inspired genetic algorithm for workflow healthcare application scheduling with hard and soft deadline constraints in hybrid clouds. Applied Soft Computing 128: 109440. https://doi.org/10.1016/j.asoc.2022.109440

[23] Karami S, Azizi S, Ahmadizar F (2024) A bi-objective workflow scheduling in virtualized fog-cloud computing using NSGA-II with semi-greedy initialization. Applied Soft Computing 151: 111142. https://doi.org/10.1016/j.asoc.2023.111142

[24] Xia Y, Luo X, Yang W, et al (2024) Dynamic variable analysis guided adaptive evolutionary multi-objective scheduling for large-scale workflows in cloud computing. Swarm and Evolutionary Computation 90: 101654. https://doi.org/10.1016/j.swevo.2024.101654

[25] Xue F, Wen J, Wang P, et al (2025) A decomposition-based multi-objective evolutionary algorithm with reinforcement learning for workflow scheduling in cloud computing environment. Cluster Computing 28(10): 678. https://doi.org/10.1007/s10586-025-05369-y

[26] Qiu H, Xia X, Li Y, et al (2023) A dynamic multipopulation genetic algorithm for multiobjective workflow scheduling based on the longest common sequence. Swarm and Evolutionary Computation 78: 101291. https://doi.org/10.1016/j.swevo.2023.101291

[27] Mangalampalli S, Hashmi S S, Gupta A, et al (2024) Multi Objective Prioritized Workflow Scheduling Using Deep Reinforcement Based Learning in Cloud Computing. IEEE Access 12: 5373-5392. https://doi.org/10.1109/access.2024.3350741

[28] Wu D, Wang X, Wang X, et al (2024) Multi-objective optimization-based workflow scheduling for applications with data locality and deadline constraints in geo-distributed clouds. Future Generation Computer Systems 157: 485-498. https://doi.org/10.1016/j.future.2024.04.004

[29] Jalali Khalil Abadi Z, Mansouri N, Javidi M M (2024) Deep reinforcement learning-based scheduling in distributed systems: a critical review. Knowledge and Information Systems 66(10):5709–5782. https://doi.org/10.1007/s10115-024-02167-7

[30] Uma K M, Shukla S (2025) Energy and performance-aware workflow scheduler using dynamic virtual network resource optimization under edge-cloud platform. Computers and Electrical Engineering 123: 110085. https://doi.org/10.1016/j.compeleceng.2025.110085

[31] Singhal S, Sharma A (2025) An energy-aware migration framework using metaheuristic algorithm in cloud computing. Knowledge and Information Systems 67(2):1373–1398. https://doi.org/10.1007/s10115-024-02224-1

[32] Tang X, Liu F, Wang B, et al (2024) Workflow scheduling based on asynchronous advantage actor–critic algorithm in multi-cloud environment. Expert Systems with Applications 258: 125245. https://doi.org/10.1016/j.eswa.2024.125245

[33] Ye L, Yang L, Zhao X, et al (2026) Energy-minimized scheduling for reliable workflow applications in heterogeneous cloud computing systems. Information Sciences 730: 122856. https://doi.org/10.1016/j.ins.2025.122856

[34] Girgis M R, Mahmoud T M, Azzam H M (2024) GA-based QOS-aware workflow scheduling of deadline tasks in grid computing. Knowledge and Information Systems 66(5):2859–2884. https://doi.org/10.1007/s10115-023-02048-5

[35] Hussaini S M, Razak T A, Jamil M A (2025) Behavior of SPEA2 Algorithm to Resolve Scheduling Problem for IoT Cloud. Journal of Intelligent Systems and Internet of Things 15(2): 138–150. https://doi.org/10.54216/jisiot.150210

[36] Zhang G, Zhang A, Sun C, et al (2025) An Evolutionary Algorithm for Multi-Objective Workflow Scheduling with Adaptive Dynamic Grouping. Electronics 14(13): 2586. https://doi.org/10.3390/electronics14132586

[37] Samti A Y, Ben Jaafar I, Nouaouri I, et al (2025) A Novel NSGA-III-GKM++ Framework for Multi-Objective Cloud Resource Brokerage Optimization. Mathematics 13(13): 2042. https://doi.org/10.3390/math13132042

[38] Amazon Web Services (2026) Amazon EC2 On-Demand Pricing. https://aws.amazon.com/ec2/pricing/on-demand/.