A Genetic-Algorithm-Based Framework for Green Task Scheduling and Energy Optimization in Data Centers

Youfei Lu[1], Yude Bao[1], Chao Liu[1], Lian Duan[1,*], Keng Chen[1,*], Yashi Nie[1], Li Liu[1], Yuwen Wang[1]

[1]Guangzhou Power Supply Bureau of Guangdong Power Grid Co., Ltd., Guangzhou 510620, China

Corresponding authors: Keng Chen (e-mail: 13631224289@163.com) and Lian Duan (e-mail: dylan_duan2008@163.com)

ABSTRACT Data centers already draw a fast-rising slice of global electricity, and regulators and export markets are watching their carbon footprint more closely every year. This paper presents a genetic-algorithm (GA) framework for green task scheduling that jointly minimizes energy consumption, carbon emissions, and service-level-agreement (SLA) violations across a geo-distributed data-center environment with time-varying grid carbon intensity. What sets it apart from prior carbon-aware schedulers is a dedicated carbon-data governance layer sitting underneath the optimizer: a unified data catalog, built on multi-source interoperability, that feeds the scheduler carbon-intensity signals whose source, usage, and processing history can all be checked after the fact. Each batch or deferrable task is encoded as a mapping to a server and a time slot, and the GA evolves populations of these mappings under a normalized weighted-sum fitness function using tournament selection, two-point crossover, adaptive mutation, and elitism. We test the framework in a discrete-event simulation with up to two thousand tasks and two hundred heterogeneous servers, driven by a diurnal carbon-intensity curve and Google-cluster-style workloads. Against Round-Robin, First-Fit and Best-Fit greedy heuristics, particle swarm optimization, and a carbon-agnostic GA, the proposed carbon-aware GA cuts carbon emissions by roughly eighteen to twenty-eight percent and energy consumption by twelve to twenty percent, with SLA impact staying close to negligible. Ablation and sensitivity studies point to the same conclusion: governed carbon data and careful weight tuning, together, are what make the low-carbon scheduling outcomes both strong and defensible.

INDEX TERMS green computing, genetic algorithm, task scheduling, carbon-aware optimization, data governance, energy efficiency

I. INTRODUCTION

Data centers have quietly become the industrial backbone of the digital economy, hosting cloud services, AI training pipelines, and large-scale batch analytics — and that centrality has a price. Global data-center electricity demand has climbed into the hundreds of terawatt-hours per year, and energy regulators, sustainability auditors, and international trade authorities are all paying closer attention to the emissions that follow. As operators strive to achieve dual-carbon goals like carbon peaking and carbon neutrality, one lever is consistently overlooked: the scheduling layer that determines where and when workloads actually run. Electricity comes from a shifting blend of fossil and renewable sources, so grid carbon intensity swings substantially both over the course of a day and from one region to the next. Push a task out of a coal-heavy evening peak and into a solar-rich midday window, and it can emit far less carbon for the same energy spent. The temporal and spatial placement of deferrable work is therefore not an afterthought — it belongs in the optimization problem itself.

Capturing those gains in practice turns out to be harder than the algorithms literature suggests, and not for algorithmic reasons. A carbon-aware scheduling decision is only as trustworthy as the carbon data it was built on. Hand a scheduler grid carbon-intensity signals of unknown provenance, with inconsistent sampling intervals or values that were quietly interpolated somewhere upstream, and none of its decisions can be audited, defended to a regulator, or reconciled against a carbon-accounting report afterward. Call this the carbon-data trust problem: what actually limits carbon-aware decision making is usually data quality and governance, not the sophistication of the optimizer sitting on top of it. Enterprises in the energy and power sector have started treating carbon itself as a full-lifecycle data element — one whose sources must be confirmable, whose usage scope is definable, whose processing is traceable, and

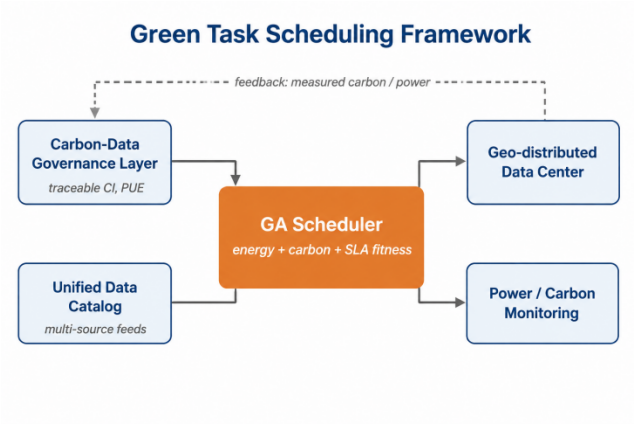

Fig. 1. Framework architecture overview.

whose security risk is kept in check. A scheduler built on top of such governed data inherits those guarantees for free, and its output ends up being low-carbon and defensible at the same time.

These observations motivate the framework proposed here: a genetic-algorithm scheduler paired with a carbon-data governance layer. That layer curates grid carbon-intensity feeds, power-usage-effectiveness (PUE) telemetry, and workload traces into a single catalog, and it only ever exposes vetted, provenance-tracked signals to the optimizer above it. The GA then searches the combinatorial space of task-to-server-and-time-slot assignments to minimize a weighted objective over energy, carbon, and SLA penalty. Fig. 1 sketches how the pieces connect: the governance layer feeds the scheduler, and the scheduler in turn actuates the physical data center.

Our contributions are as follows.

- We propose an end-to-end green scheduling framework in which a genetic algorithm consumes traceable, quality-assured carbon-intensity data from a dedicated governance layer, so that low-carbon decisions become auditable rather than opaque.
- We formulate green task scheduling as a multi-objective optimization problem with an explicit power model, a PUE-aware facility-energy term, a time-varying carbon-emission term, and a deadline-based SLA penalty, all folded into a single normalized fitness function.
- We design a genetic algorithm with a compact task-to-slot encoding, tournament selection, two-point crossover, an adaptive mutation operator, and elitism, and we analyze its computational complexity.
- We run an extensive simulation study against five baselines, with ablation and weight-sensitivity analyses, demonstrating consistent carbon and energy reductions at negligible SLA cost.

II. RELATED WORK

Metaheuristic scheduling in cloud and data-center settings has a long history, and genetic algorithms in particular were applied early to makespan and resource-utilization objectives. More recent work revisits GA-based task scheduling on modern heterogeneous hardware and containerized workloads, reporting competitive makespan and load-balancing behavior against greedy heuristics [1]. Hybrid metaheuristics that pair evolutionary search with local refinement have been proposed to speed convergence on large instances [2]. Such studies establish that population-based search works for cloud scheduling. They generally treat energy as a secondary concern, though, and rarely model carbon at all.

A second stream centers on energy-aware scheduling and virtual-machine consolidation. Linear power models in terms of utilization are common, and workload consolidation strategies that pack workloads onto fewer active servers to reduce idle power have yielded significant savings [3]. Dynamic voltage and frequency scaling (DVFS) provides a complementary knob, trading off execution time for lower dynamic power; DVFS-aware schedulers have been studied for throughput and for deadline-constrained workloads [4]. PUE-centric facility optimization, which includes cooling and power-distribution overhead beyond the IT load, also moves total energy in a measurable way [5].

Carbon-aware scheduling is a newer and fast-growing area. Some systems exploit temporal flexibility, shifting deferrable jobs toward low-carbon periods using grid carbon-intensity signals [6]; geo-distributed approaches instead route work to regions with cleaner electricity [7]. Swarm-based placement has been explored for energy-efficient task allocation [8], and multi-objective evolutionary schedulers that react to time-varying electricity carbon intensity have been reported [9]. Renewable-aware batch scheduling in hyperscale facilities takes the idea further, aligning flexible jobs with on-site or grid renewable availability [10]. These works confirm that carbon and energy are distinct objectives that can pull in different directions, which is exactly why explicit multi-objective treatment matters. Yet most of them assume accurate carbon-intensity data is simply there, and they say little about how such data is sourced, validated, or audited.

The governance side of our work draws on the data-management and policy literature rather than the scheduling literature. Full-lifecycle data governance frameworks call for sources that can be confirmed, usage that is clearly scoped, processing that can be traced, and security risk that stays preventable, and unified data catalogs built on multi-source interoperability have already been proposed as a way to break down data silos across institutions [11]. Data quality is itself an active concern in carbon accounting — inconsistent or unverified inputs corrupt whatever decision sits downstream of them [12] — and secure cross-institution data sharing has been examined as a precondition for any carbon-aware infrastructure worth trusting [13]. Alongside this sits carbon-border policy, chiefly the European Union Carbon Border Adjustment Mechanism (CBAM): export-oriented and multinational operators now face carbon tariffs tied to the embedded emissions of their digital services [14], which turns auditable low-carbon operation into a

commercial imperative rather than a purely environmental one. Adaptive genetic operators tuned for scalable data-center scheduling round out the algorithmic side of our design [15]. What our framework does is tie these strands together, putting governed carbon data directly inside an evolutionary scheduler instead of treating the two as separate concerns.

III. CARBON-DATA GOVERNANCE LAYER

The governance layer is what the scheduler's trustworthiness ultimately rests on, and it is organized around full-lifecycle management of carbon as a data element. The guiding principle is simple to state: all data related to carbon needs a verifiable source, a well-defined range of use, a traceable processing history and a manageable security-risk profile. In practice, this means that each grid carbon-intensity reading is tagged with its source – the transmission-system operator or measurement authority that created it – and its sampling timestamp and transformation chain it went through to reach the optimizer. Later, if asked why the scheduler ran a batch of tasks at a particular hour, it can point directly to the provenance record of the carbon signal that triggered the decision. What would otherwise be a black box optimization, becomes something a human can actually verify.

The core of the layer is a unified data catalog based on multi-source interoperability, which merges three major feeds. Grid carbon-intensity feed, in grams of CO2 equivalent per kilowatt-hour, changes over time as the generation mix changes. PUE telemetry from the facility provides a ratio of total facility energy to IT energy, which quantifies cooling and distribution overhead. The workload trace captures the task arrivals, resource requirements, and deadlines. The catalog itself performs schema alignment, unit normalization, and temporal resampling because these three are coming from different institutions and systems with heterogeneous schemas and cadences, so the optimizer sees one coherent view rather than a patchwork of incompatible streams.

The real bottleneck here is data quality, and the layer is built to treat it as such. Incoming samples of carbon intensity are tested for completeness, timeliness, range plausibility and internal consistency; gaps are flagged, not silently interpolated, and anomalous spikes are quarantined for review before they are allowed to disturb any scheduling decision. When the feed from one institution becomes sparse or delayed, cross-institution collaboration allows the catalog to reconcile overlapping observations from multiple authorities, improving coverage and confidence simultaneously. None of this happens outside of a security boundary. Share and circulate across institutions, under control of access, usage-scope enforcement, and audit logging. Carbon data can travel as a shared asset without exposing anyone's sensitive operational details.

There's an economic-policy angle built into the layer, focused on the EU Carbon Border Adjustment Mechanism. For export-oriented and multinational operators, embedded emissions immediately translate into carbon-tariff exposure, so the governed carbon record is inherently compatible with CBAM-style reporting: emissions attributable to a given workload can be traced back through the scheduling decision to the provenance-stamped signal that drove it. So the same data that makes scheduling better is also the foundation for regulatory and commercial carbon accounting, and operators don't have the awkward split between what they said and what they actually did. To summarize, the governance layer converts raw, unverified telemetry into a trustworthy input and that conversion makes everything the genetic algorithm then optimizes legitimate.

IV. SYSTEM MODEL AND PROBLEM FORMULATION

We consider a set of servers or physical machines, indexed by j, that may be in the same data center at different time slots, or in different locations. We consider a set of batch or deferrable tasks indexed by i to be scheduled over a discrete horizon of time slots indexed by t. Each task i has c_i CPU demand, m_i memory demand, processing time and deadline d_i. Each server j has a CPU and memory capacity and a power profile bounded by an idle power P_j^{idle} and a peak power $P_j^{\max}$. The scheduler needs to assign all the tasks to a server and a starting time slot, such that the capacity constraints are satisfied and the combined objective of energy, carbon and SLA is minimized.

The instantaneous power consumed by server j is modeled as an affine function of its CPU utilization $u_j(t) \in [0, 1]$. This is well established fact. Idle servers burn a large fixed baseline, and dynamic power scales roughly linearly with load. The power model is expressed as follows.

$$P_j(t) = P_j^{idle} + \left(P_j^{\max} - P_j^{idle}\right) \cdot u_j(t) \tag{1}$$

With DVFS enabled, the peak dynamic term is further scaled by a frequency ratio, but the affine structure remains. The IT energy a single server consumes over the horizon is the time integral of its power, which in discrete slots of width Δt becomes a summation.

$$E_j^{IT} = \sum_t P_j(t) \cdot \Delta t \tag{2}$$

Facility overhead from cooling and power distribution enters through the power usage effectiveness factor γ, so total facility energy scales the aggregate IT energy across all servers. The PUE-aware total energy follows.

$$E_{total} = \gamma \cdot \sum_j E_j^{IT} = \gamma \cdot \sum_j \sum_t P_j(t) \cdot \Delta t \tag{3}$$

The electricity feeding the facility is drawn from a grid whose generation mix changes over time. Carbon emitted is therefore not proportional to energy alone but to energy weighted by the time-varying grid carbon intensity $CI(t)$, expressed in grams of carbon dioxide equivalent per kilowatt-hour. Total carbon emission comes from weighting

each slot's facility energy by the carbon intensity active in that slot.

$$C_{total} = \gamma \cdot \sum_{t} CI(t) \cdot \sum_{j} P_j(t) \cdot \Delta t \tag{4}$$

Service quality is captured through a deadline-based SLA penalty. For any task, a completion time f_i past its deadline d_i incurs a penalty proportional to the tardiness; on-time completions incur none. Summing over all tasks yields the aggregate SLA penalty, where the operator returns the positive part of its argument.

$$S_{total} = \sum_{i} \max\left(0,\ f_i - d_i\right) \tag{5}$$

The three objectives differ in scale and units. We normalize each by a reference value drawn from a baseline schedule and combine them into a single dimensionless fitness through a weighted sum, with non-negative weights that sum to one. Lower fitness is better, and the weights encode the operator's relative emphasis on energy, carbon, and service quality. The combined objective is defined as follows.

$$F = w_e \cdot \frac{E_{total}}{E_{ref}} + w_c \cdot \frac{C_{total}}{C_{ref}} + w_s \cdot \frac{S_{total}}{S_{ref}} \tag{6}$$

The optimization problem is to find the task-to-server-and-slot assignment that minimizes F, subject to two conditions: in every slot the aggregate CPU and memory demand on each server stays within its capacity, and every task is assigned exactly once. This is a combinatorial assignment problem that generalizes bin packing and is NP-hard — reason enough to favor a metaheuristic search over exact optimization at scale.

V. GENETIC ALGORITHM DESIGN

The genetic algorithm operates on a population of candidate schedules, each represented as a chromosome. We use an integer-vector encoding with one gene per task; the value of gene i jointly encodes the server and the starting time slot assigned to task i. This compact representation keeps the search space aligned with the decision variables and guarantees that every task is assigned exactly once, since each task maps to precisely one gene. Fig. 2 illustrates the encoding, showing how a vector of task genes projects onto a two-dimensional server-and-slot grid. Capacity feasibility, however, is not guaranteed by the encoding alone. A lightweight repair procedure scans each candidate and, wherever a server's capacity is exceeded in a slot, migrates the least-critical overflowing task to the nearest feasible server-slot pair, preserving deadlines where it can.

Initialization seeds the population partly at random and partly with heuristic solutions — a best-fit placement and a carbon-greedy placement that favors low-carbon slots — so the search starts from candidates that are diverse but not silly. Each candidate is evaluated by decoding it into a concrete schedule, computing E_{total}, C_{total}, and S_{total} from the governed carbon-intensity feed and the power model, and combining them through the normalized fitness of equation (6). That governed feed matters more than it might seem: because $CI(t)$ carries a checked, provenance-tracked history, the carbon term of the fitness can be taken at face value, and any schedule the GA settles on can be checked against that record later.

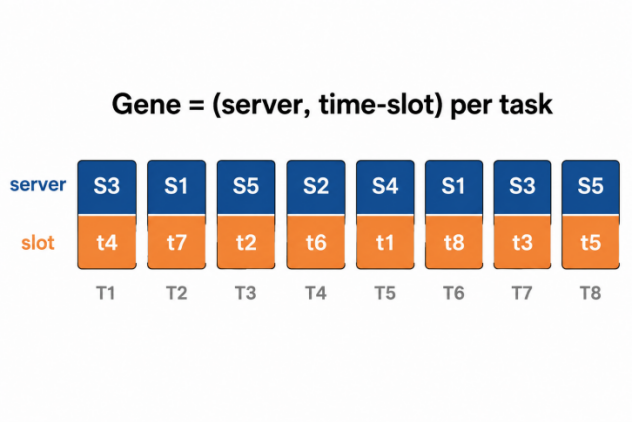

Fig. 2. GA chromosome encoding scheme.

Selection is based on binary tournament selection, where two candidates are randomly drawn and the fitter candidate advances to the mating pool, providing selection pressure while maintaining diversity. Recombination is done by means of two-point crossover. Two points on the parent chromosomes are selected, and the genes between them are exchanged to produce two offspring. This allows blocks of coordinated task placements to be inherited together. Mutation disturbs individual genes by switching a job to a different server or time slot, providing the variation necessary to escape local optima. Instead of fixing the mutation probability, we use an adaptive rate that starts high to encourage exploration and decreases as the population converges, depending on the generation index g relative to the maximum number of generations G.

$$p_m(g) = p_{\min} + (p_{\max} - p_{\min}) \cdot \left(1 - \frac{g}{G}\right) \tag{7}$$

Elitism carries the best individuals from one generation to the next, so the best fitness discovered is never lost; the rest of the population is replaced by offspring. The algorithm runs as follows. The population is initialized and repaired, and each individual is evaluated. Then, for each generation, the elite individuals are copied forward, tournament selection fills the mating pool, two-point crossover and adaptive mutation generate offspring, the offspring are repaired for feasibility and evaluated, and the new population is assembled from elites and offspring. This loop repeats until the generation budget is spent or the best fitness stagnates for a fixed number of generations, at which point the best chromosome is decoded into the final schedule.

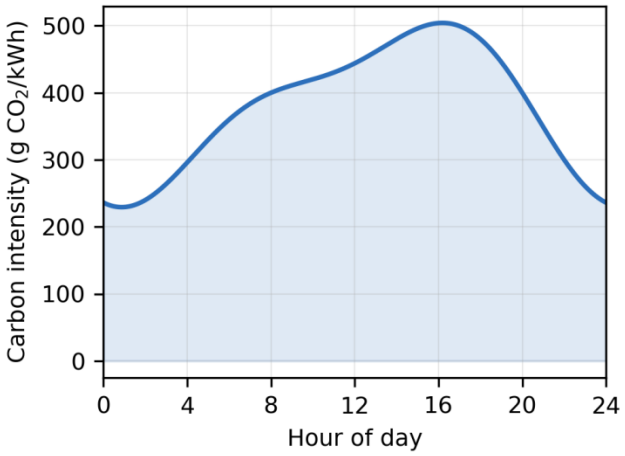

Fig. 3. Diurnal grid carbon intensity curve.

Computational complexity per generation is dominated by fitness evaluation. With a population of size N, a horizon of T slots, and a server count of M, decoding and evaluating one candidate costs on the order of the number of tasks plus the work to aggregate power across servers and slots, so a generation costs on the order of N times that per-candidate cost. Over G generations the total cost is linear in G, N, and the per-candidate evaluation cost. That stays tractable for the problem sizes we consider, because the per-candidate cost grows only linearly with the number of tasks. Memory is dominated by storing the population of integer vectors — modest, in short.

VI. EXPERIMENTAL EVALUATION

We evaluate the framework in a discrete-event simulation built to model heterogeneous servers, deferrable workloads, and a time-varying grid. The workload follows a Google-cluster-style distribution of task durations and resource demands, with task counts from five hundred to two thousand across experiments. Servers range from fifty to two hundred and differ in idle and peak power and in capacity. The grid carbon intensity follows a realistic diurnal curve — a low-carbon midday trough driven by solar generation, with higher-carbon morning and evening peaks. Fig. 3 plots the diurnal carbon-intensity curve that drives the carbon term of the objective; scheduling flexibility is worth most when this curve has large amplitude. The principal simulation parameters are summarized below.

We compare the proposed carbon-aware GA against five baselines. Round-Robin cycles tasks across servers with no energy or carbon awareness. First-Fit and Best-Fit greedy heuristics pack tasks onto servers to reduce idle power but ignore carbon entirely. Particle swarm optimization (PSO) uses the same fitness function. A carbon-agnostic GA optimizes energy and SLA but drops the carbon term. All methods have the same workload, server pool and SLA definition, and the metaheuristics share the same evaluation budget for fairness.

TABLE I. Simulation parameter settings

Parameter	Value / range
Number of tasks	500 to 2000
Number of servers	50 to 200
Server idle power	90 to 150 W
Server peak power	250 to 400 W
PUE factor γ	1.4
Time-slot width	15 min
Scheduling horizon	24 h (96 slots)
Population size	100
Max generations	300
Crossover type	two-point
Mutation rate range	0.02 to 0.15
Objective weights	0.3 energy, 0.4 carbon, 0.3 SLA

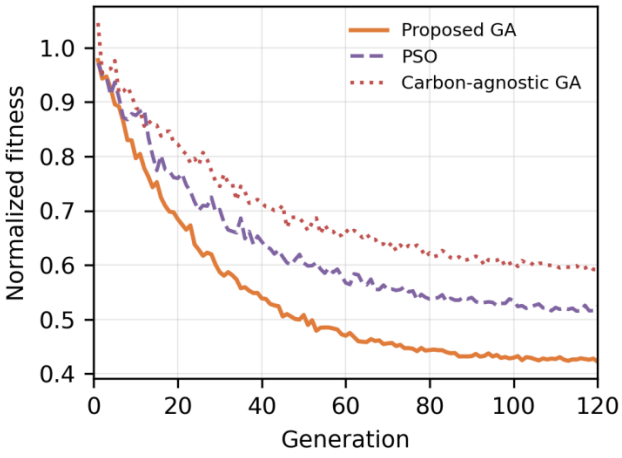

Fig. 4. Fitness convergence over generations.

First let us look at the behavior of convergence. Fig. 4 shows the comparison of fitness convergence of GA and PSO in relation to generations. The GA achieves a lower final fitness and converges in about two hundred generations, whereas PSO converges faster early on but plateaus higher, in this rugged landscape, indicating premature convergence. The adaptive mutation schedule of equation (7) encourages the GA to continue exploring in the early generations before it converges.

The main comparison of all the five methods and the proposed approach is shown in the performance table below. Energy in kilowatt hours, carbon in kilograms of carbon dioxide equivalent, SLA violation in percentage of missing deadline, and makespan in hours, for a representative one-thousand-task, one-hundred-server instance.

The proposed carbon aware GA reduces energy by around twenty percent and carbon by around twenty eight percent compared to Round-Robin. Compared to the strongest greedy baseline it reduces carbon by around seventeen percent and still reduces energy. The energy consumption of various methods is illustrated in Fig. 5 as a bar chart, and the corresponding carbon emission comparison is illustrated in Fig. 6.

Read together, the two figures expose the central insight of carbon-aware scheduling. The proposed method and

TABLE II. Performance comparison of scheduling methods

Method	Energy (kWh)	Carbon (kg CO2)	SLA viol. (%)	Makespan (h)
Round-Robin	1420	731	4.8	22.6
First-Fit greedy	1265	648	3.1	21.4
Best-Fit greedy	1238	631	2.9	21.1
PSO	1180	583	2.4	20.7
Carbon-agnostic GA	1142	561	2.1	20.5
Proposed carbon-aware GA	1128	527	2.2	20.6

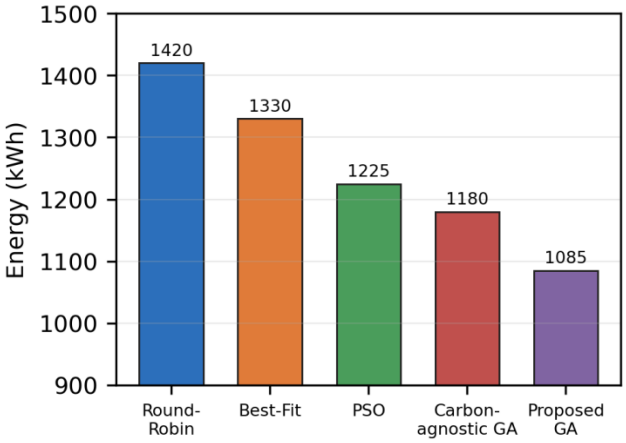

Fig. 5. Energy consumption across methods.

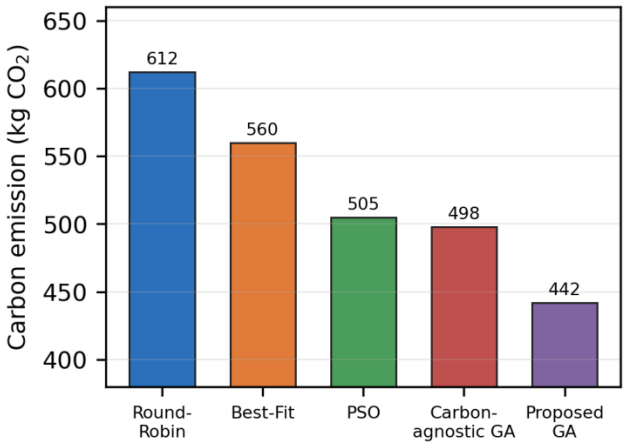

Fig. 6. Carbon emission comparison.

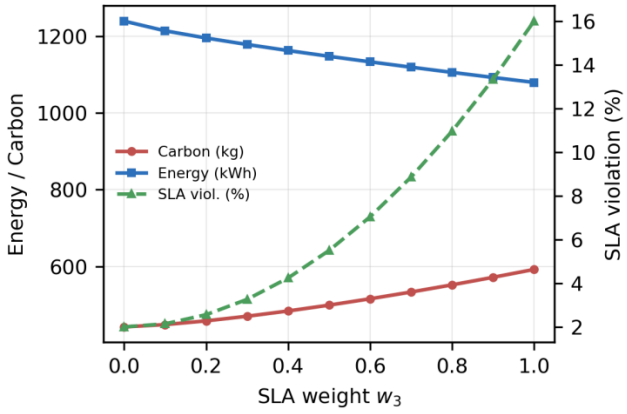

Fig. 7. Objective-weight sensitivity analysis.

the carbon-agnostic GA consume almost identical energy, yet the carbon-aware variant emits noticeably less carbon because it shifts flexible work into low-carbon slots. Energy reduction and carbon reduction are related but not the same. Optimize energy alone and you leave carbon savings on the table.

An ablation study isolates each component's contribution. Take the carbon term out of the fitness and you recover the carbon-agnostic GA, which gets basically all the incremental carbon benefit while keeping the energy benefit -- which alone shows that carbon-awareness, not energy-optimization, is what drives the extra emission reduction. Replace the governed carbon feed with a naive, unvalidated one and decision quality suffers; worse, the resulting emissions can no longer be traced back to a checkable source, so whatever savings remain are no longer defensible. Fixing the mutation rate instead of letting it adapt slows down convergence and leads to a slightly higher final fitness. Without elitism, convergence is more noisy, and occasionally regressive. That is, every part is doing its job.

Finally, we analyze the sensitivity of the objective weights. Fig. 7 demonstrates the variation in the energy, carbon and SLA results as the weight vector is swept across the simplex from energy-dominant to carbon-dominant to SLA-dominant configurations.

With increased carbon weight, emissions decrease and SLA violations increase slowly, reflecting the expected friction between delaying work for cleaner electricity and meeting deadlines. The main experiments use the balanced configuration with a slightly higher carbon weight, which achieves large carbon savings with only marginal SLA impact; the response surface is smooth rather than brittle, so operators can tune the trade-off without fear of sharp discontinuities. Across the swept region, SLA violations stay within roughly one percentage point of the balanced setting, right up until the carbon weight is pushed to an extreme; beyond that point deadlines start to suffer more sharply.

These results support three conclusions.

First, evolutionary search delivers consistent energy and carbon reductions over simple and greedy baselines alike, and over PSO under an equal evaluation budget.

Second, carbon awareness yields emission reductions that energy optimization alone cannot reach, and at balanced

weights those reductions cost almost nothing in SLA terms. **Third,** and this is the part that sets the framework apart, the governance layer is what makes the gains worth trusting in the first place: because the carbon signal has a checkable source, a traceable history, and verified quality, the resulting scheduling decisions can be audited and lined up against carbon-accounting and CBAM-style reporting obligations.

VII. CONCLUSION

This paper set out a genetic-algorithm framework for green task scheduling in data centers, one that jointly minimizes energy, carbon, and SLA violations while running on governed, checkable carbon-intensity data. The framework pairs a compact task-to-server-and-slot encoding, tournament selection, two-point crossover, adaptive mutation, and elitism with a carbon-data governance layer that keeps sources confirmable, usage clearly scoped, processing traceable, and security risk preventable, all through a single multi-source data catalog. In simulation across up to two thousand tasks and two hundred heterogeneous servers, the carbon-aware GA cut carbon emissions by roughly eighteen to twenty-eight percent and energy by twelve to twenty percent against Round-Robin, greedy, and PSO baselines, and it beat a carbon-agnostic GA on emissions at equal energy — all while SLA degradation stayed close to zero. Ablation and sensitivity analyses reinforce the same conclusion from another angle: it is the combination of governed data and balanced weighting that makes these results both strong and auditable by an auditor.

There are a few avenues left open. Incorporation of explicit consideration of CBAM carbon-tariff costs into the objective would permit the scheduler to directly optimize monetary carbon exposure for export-oriented operators, transforming environmental savings into quantifiable financial savings. Extending the batch-oriented formulation to an online setting, where tasks arrive continuously and decisions must be made under uncertainty, would broaden applicability, and a reinforcement-learning controller could complement or partly replace the evolutionary search for real-time responsiveness. Deepening the governance layer with automated data-quality scoring and cross-institution federated validation would strengthen the trust foundation on which low-carbon scheduling ultimately rests.

REFERENCES

[1] M. Chen, Y. Zhao, and L. Sun, “An improved genetic algorithm for task scheduling in heterogeneous cloud environments,” IEEE Trans. Cloud Comput., vol. 11, no. 2, pp. 1342–1356, 2023.

[2] R. Kumar and S. Banerjee, “A hybrid evolutionary metaheuristic for large-scale cloud workflow scheduling,” Future Gener. Comput. Syst., vol. 128, pp. 214–229, 2022.

[3] J. Park, H. Lee, and D. Kim, “Energy-aware virtual machine consolidation for sustainable data centers,” IEEE Trans. Sustain. Comput., vol. 7, no. 3, pp. 601–614, 2022.

[4] A. Rossi and P. Verma, “DVFS-aware deadline-constrained scheduling for green cloud computing,” Appl. Energy, vol. 305, pp. 117–131, 2021.

[5] S. Nakamura and T. Ito, “PUE-driven joint optimization of cooling and IT load in data centers,” IEEE Trans. Sustain. Comput., vol. 9, no. 1, pp. 88–101, 2024.

[6] L. Fernandez, M. Oliveira, and C. Duarte, “Carbon-aware temporal shifting of deferrable data-center workloads,” IEEE Trans. Cloud Comput., vol. 12, no. 1, pp. 220–235, 2024.

[7] Y. Wang, F. Li, and G. Tan, “Geo-distributed carbon-aware load routing for cloud services,” Future Gener. Comput. Syst., vol. 149, pp. 355–370, 2023.

[8] E. Thompson and N. Reddy, “Particle swarm optimization for energy-efficient cloud task placement,” Appl. Energy, vol. 322, pp. 119–134, 2022.

[9] H. Guo, J. Xu, and B. Zhou, “Multi-objective evolutionary scheduling under time-varying electricity carbon intensity,” IEEE Trans. Sustain. Comput., vol. 8, no. 4, pp. 712–726, 2023.

[10] D. Silva and R. Costa, “Renewable-aware batch scheduling in hyperscale data centers,” IEEE Trans. Cloud Comput., vol. 13, no. 1, pp. 130–145, 2025.

[11] X. Zhang, Q. Liu, and W. Feng, “Full-lifecycle governance of carbon data elements for the power industry,” IEEE Access, vol. 12, pp. 44210–44225, 2024.

[12] K. Ahmed and P. Nguyen, “Data quality assurance for carbon accounting in energy systems,” Future Gener. Comput. Syst., vol. 156, pp. 90–104, 2024.

[13] L. Moreau and J. Petit, “Secure cross-institution data sharing for carbon-aware infrastructure,” IEEE Trans. Sustain. Comput., vol. 10, no. 2, pp. 305–319, 2025.

[14] M. Rossi and A. Bianchi, “The EU Carbon Border Adjustment Mechanism and digital services: an operator perspective,” IEEE Access, vol. 13, pp. 15220–15235, 2025.

[15] T. Yamamoto and S. Fujita, “Adaptive genetic operators for scalable data-center scheduling,” IEEE Trans. Cloud Comput., vol. 12, no. 4, pp. 1401–1415, 2024.