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.

# An Adaptive Grey Wolf Optimizer for Energy-Aware Virtual Machine Consolidation in Sustainable Green Data Centers

**Maad khalil**

maad.khalil@uodiyala.edu.iq

University of Diyala

**Odiljan Turdiyev**

Tashkent State Transport University

**Ibtesam Hawi**

University of Diyala

**Tolaniddin Nurmukhamedov**

Tashkent State Transport University

**Research Article**

**Keywords:** virtual machine consolidation, grey wolf optimizer, adaptive metaheuristics, green cloud computing, energy efficiency, SLA management, CloudSim Plus

**Posted Date:** September 30th, 2026

**DOI:** https://doi.org/10.21203/rs.3.rs-11062185/v1

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

**Additional Declarations:** No competing interests reported.

# An Adaptive Grey Wolf Optimizer for Energy-Aware Virtual Machine Consolidation in Sustainable Green Data Centers

**Abstract —** Global data center energy demand has increased by about 10% per year for the last 10 years, and is projected to account for 3–4% of global electricity consumption by 2030. Virtual Machine Consolidation (VMC), i.e. packing workloads onto a minimal set of active physical hosts, is the most tractable path to cut this footprint, but the underlying placement problem is NP-Hard and grows combinatorially with fleet size. Modern cloud infrastructures are often characterised by high-dimensional, non-stationary workload environments where conventional metaheuristic solvers, like the widely used Grey Wolf Optimiser (GWO), employ a linear decaying control parameter, which results in premature exploitation. In this paper, we present AGWO-VMC (Adaptive Grey Wolf Optimiser), where the classical control parameter a has been substituted by a gradient-feedback decay function which continuously tracks the trajectory of the fitness progress and modulates the exploration-exploitation balance.

We adopt three complementary innovations, i.e. non-linear adaptive decay, probabilistic discrete encoding, and a periodic 2-Opt local search on the leading solution, in a multi-objective fitness function including server energy, SLA violations, migration overhead and load imbalance. Experiments in CloudSim Plus with the Google Cluster Trace (2019) and the Azure Public Dataset (2024) on 800-1000 heterogeneous physical machines demonstrate that for four workload scenarios (Static, Dynamic Steady, Flash Crowd, Heterogeneous Mix), AGWO-VMC reduces the energy consumption relative to standard GWO by 18.5-23.7% depending on scenario (up to 29.3% relative to PSO and up to 32.8% relative to GA), while maintaining composite SLA violation rates below 1.9% in all cases. The ablation experiments verify that the adaptive decay mechanism itself contributes to 21.4 percentage points of the energy improvement. Wilcoxon signed-rank tests show $p<0.001$ and large effect size (Cohen's $d>0.8$) for all pairs of scenarios and algorithms. The implementation is released as open-source software to be used as a reproducible benchmark for green data center research.

**Keywords:** virtual machine consolidation; grey wolf optimizer; adaptive metaheuristics; green cloud computing; energy efficiency; SLA management; CloudSim Plus.

## 1. Introduction

The exponential growth of cloud-native workloads such as large language model inference, real-time video analytics and financial trading platforms has driven the global data center electricity consumption sharply upwards. The International Energy Agency estimates that by 2030 data centers, along with the broader artificial-intelligence and cryptocurrency-mining workloads they increasingly are supporting, will account for 3-4% of global electricity consumption [1]. Demand has increased and so has capacity, but there is a structural inefficiency built in: physical servers in operational data centers run at an average of 15-20% of their rated

CPU capacity, wasting energy on idle circuitry that still draws 50-60% of peak power. This utilisation gap is addressed by an immediate and most economical approach, namely software-enabled task consolidation.

This is where Virtual Machine Consolidation (VMC) plays a role by alleviating this problem by dynamically migrating VMs to the least number of PMs in operation and switching off the rest. Empirical implementations have demonstrated energy reductions of over 30% against non-consolidated baselines [3]. The difficulty is that VMC is NP-Hard in general: allocating n VMs to m PMs with multi-dimensional capacity constraints is equivalent to multi-dimensional bin packing, and the number of fleet sizes in hyperscale clouds is often more than tens of thousands of nodes. In addition to the combinatorial difficulty, each placement selection faces two conflicting goals. Aggressive consolidation leads to concentration of CPU demand on a few hosts which increases the probability of SLA violations due to resource contention. However, live VM migration, which is the mechanism that enables consolidation, has its own energy cost as the state of RAM is transferred over the network [4]. So a good VMC algorithm must strike a balance between the competing needs of energy minimisation, service quality and migration cost.

Population metaheuristic algorithms have become the leading technique for large-scale VMC, giving near-optimal solutions within acceptable operational times. Here is a very suitable scenario for the Grey Wolf Optimiser (GWO) introduced by Mirjalili et al. [5]. It has a four-level social hierarchy (α, β, δ, ω) that applies natural exploitation pressure through leader-follower position changes. It has only two control parameters and has shown to be competitive with PSO, GA and Differential Evolution on discrete engineering problems [21]. In the standard GWO, the exploitation versus exploration trade-off is controlled by a scalar a(t) which decreases linearly from 2 to 0 over a fixed period of time. This fixed schedule does not match well with VMC work-load variations: during peak loads, the working environment changes abruptly, and a prematurely exploitative algorithm may get trapped in states that do not react fast enough to keep SLA-compliant. In contrast, in calm periods, over-exploration costs consolidation cycles.

Recent adaptive variants of GWO have explored chaotic map scheduling [27], fuzzy rule based adjustments [28] and opposition-based initialisation [31]. All these methods only change the parameter trajectory based on the number of iterations or the population statistics. None of these methodologies directly observes the optimisation landscape through a problem-specific signal. In the VMC framework, the most informative signal is the rate of fitness improvement per consolidation cycle: if this gradient tends to zero, the algorithm is stuck in a local optimum and should widen its search; if it is significantly positive, the algorithm is on a profitable track and should intensify exploitation. This observation governs the basic design of the proposed technique.

In this work, we present AGWO-VMC, an Adaptive Grey Wolf Optimiser for Virtual Machine Consolidation. The main innovation is a gradient-feedback decay mechanism, which redefines $a(t)$ as a function of the elapsed iteration fraction and the smoothed fitness gradient over a trailing window. When the gradient is near zero , decay is low and we continue exploring . When the gradient is strongly positive , decay is high and we are encouraged to exploit . The core mechanism is further enhanced by two design decisions: 1) a probabilistic discrete encoding scheme that maps continuous wolf positions to integer PM identifiers via a temperature-controlled softmax, eliminating the quantisation artefacts of simple rounding; and 2) a periodic 2-Opt local search applied only to the $\alpha-$wolf solution, which allows for detailed refinement of the optimal configuration while preserving diversity in the population.

This work's primary contributions are as follows:

1. A gradient-feedback adaptive decay mechanism for the control parameter of GWO, accompanied by formal boundedness and monotonicity guarantees for the resulting control-parameter trajectory

(monotone fitness non-increase, a restricted parameter range, and asymptotic exploitation as the iteration budget is exhausted), and substantiated through a specialized ablation investigation.

2. A multi-objective VMC fitness function that concurrently reduces server energy consumption, composite SLA violations (SLATAH × PDM), migration energy costs, and load imbalance, utilizing AHP-justified weights based on actual data center operations literature.
3. Extensive simulation study over four workload scenarios and 800-1,000 heterogeneous physical machines, with evaluations of six baselines (BFD, GA, PSO, vanilla GWO, NSGA-II and DRL-based consolidation), including full Wilcoxon, effect-size, and scalability analysis.
4. Five-factor ablation research that quantifies the contribution of each algorithmic component to the improvement of overall performance.
5. An open-source implementation of CloudSim Plus that offers a replicable benchmark for future study on green Virtual Machine Consolidation.

The subsequent sections of this work are structured as follows. Section 2 provides a literature review on four thematic areas. Section 3 formally defines the VMC problem. Section 4 introduces the proposed comprehensive AGWO-VMC framework, which encompasses the gradient-feedback mechanism, the discrete encoding, and the pseudocode. Section 5 provides a detailed analysis of convergence, the exploration and exploitation trade-off, and scalability. Section 6 describes the experimental setup. Section 7 presents and discusses the results. Section 8 discusses the implications and limitations, while Section 9 provides the conclusion.

## 2. Related Work

### 2.1 Virtual Machine Consolidation Algorithms

For the VMC problem, we formulated deterministic bin-packing heuristics, namely Best Fit Decreasing (BFD) and First Fit Decreasing (FFD) which are practical benchmarks with O(n log n) complexity [7]. These algorithms have a known limitation: they consider each resource parameter independently and always result in placements that are 15–25% more energy-hungry than metaheuristic approaches for diverse or time-varying workload profiles. Among the first population-based methods for VMC, genetic algorithms (GA) demonstrated that crossover and mutation operators can find consolidation configurations that are not reachable by greedy heuristics [8]. Their practical limitation is parameter sensitivity (population size, crossover rate, mutation probability need problem-specific tuning) and slow convergence in large fleets (often 200-300 generations before stabilisation). Because of the simplicity in parameterisation and the faster convergence on continuous benchmarks, PSO has attracted sustained interests. Multi-swarm variants and chaotic inertia-weight schedules have improved the performance of PSO by 15-20% over the standard formulation [10, 11]. Ant Colony Optimization (ACO) has been used for VM placement by modeling pheromone trails as markers of good host-selection patterns [9, 12]. Simulated Annealing (SA) and Artificial Bee Colony (ABC) approaches have been studied for their ability to escape local optima; however, both require careful scheduling or parameter configuration to work well in high-dimensional environments [13, 14].

Recently, reinforcement learning has been integrated into the VMC domain. Deep Q-Network (DQN) and actor-critic agents, trained on historical workload traces, can develop placement strategies that predict load

fluctuations, resulting in 12–18% higher energy savings compared to reactive heuristics in assessed installations [15]. Graph neural networks (GNNs) have been successfully used to learn placement embeddings from host topology, showing promising results on medium-sized fleets [18]. The main disadvantage of these learning-based approaches is that they depend on large training datasets that need to match the deployment distribution, with a large performance drop when used on unseen workload conditions. Hybrid frameworks combining neural fitness surrogates with metaheuristic search have been proposed to increase convergence speed; however, they come with additional hyperparameters and training costs that prevent deployment in production environments.

### 2.2 Grey Wolf Optimizer: Selection Rationale

The GWO was chosen for this task due to four complementing attributes. The hierarchical leadership structure, wherein α, β, and δ wolves direct ω wolves through weighted position updates, establishes a natural exploitation gradient that aligns with the iterative refinement framework of VMC: a limited number of high-quality consolidation configurations steer the remainder of the population towards denser placements. Secondly, GWO necessitates just two principal control parameters (a and C), hence alleviating the tuning effort compared to PSO (three parameters) and GA (three or more) in an environment where workload patterns fluctuate on sub-hour timescales. Third, a large number of studies in the literature on cloud computing support the practical superiority of GWO. For instance, the work of El-Said et al. [34] produced 15-18% energy saving for workflow scheduling in comparison to PSO, and Nadimi-Shahraki et al. [21] proved the dominance of GWO over Differential Evolution on discrete engineering benchmarks. The position update of GWO is easily extendable to discrete spaces via probabilistic mapping, while covariance-matrix adaptation methods are designed for continuous spaces..

### 2.3 Adaptive GWO Variants

Since its original presentation by Mirjalili et al. [5], the GWO literature has investigated several ways to alleviate the rigid linear decreasing nature of $\alpha(t)$. The Chaotic GWO [27] uses logistic, tent or sinusoidal maps instead of the linear schedule and produces pseudo-random sequences of parameters , which increase the performance on multimodal benchmarks by 10–15%.The fuzzy-adaptive GWO [28] regulates parameter updates based on iteration count and population fitness variance via a fuzzy rule base, resulting in similar enhancements while incurring increased complexity in rule formulation. PSO-GWO hybrids incorporate PSO velocity updates to diversify wolf locations, whereas GWO-GA hybrids implement crossover operators to evade local optima. The opposition-based GWO [31] evaluates candidate locations and their reflections simultaneously and thus doubles the effective search volume at each iteration. Lévy-flight GWO [32] substitutes conventional updates with heavy-tailed step distributions optimized for extensive exploration. Multi-objective extensions [33] integrate non-dominated sorting and crowding distance for the estimate of the Pareto front.

A significant deficiency exists among these variants: their adaptation techniques are influenced by iteration count or general population statistics, rather than by problem-specific feedback from the optimization trajectory. Different workloads change the fitness landscape in the VMC environment. The most relevant metric is the improvement rate of the consolidation quality at each decision cycle, i.e., the metric that AGWO-VMC aims to monitor and respond to. So far, to the best of the authors' knowledge, no previous work has combined GWO parameter adaptation with convergence gradient feedback in a discrete multi-objective VMC framework.

Table 0. Adaptation trigger used by prior adaptive-GWO variants, contrasted with AGWO-VMC's trajectory-based gradient feedback.

| Variant | Adaptation Trigger | Signal Source | Responds to Live Search Trajectory? |
|---|---|---|---|
| Chaotic GWO [27] | Logistic / tent / sinusoidal map | Deterministic chaotic sequence | No |
| Fuzzy-adaptive GWO [28] | Iteration count + population fitness variance | Population-level statistic | Partial (population, not trajectory) |
| Opposition-based GWO [31] | Reflection of candidates at initialization | Static, one-time transform | No |
| Lévy-flight GWO [32] | Heavy-tailed step-size distribution | Fixed probability distribution | No |
| Multi-objective GWO [33] | Non-dominated sorting + crowding distance | Population-level ranking | No |
| **AGWO-VMC (proposed)** | **Trailing fitness-improvement gradient** | **Per-iteration optimization trajectory** | **Yes** |

### 2.4 Energy Modelling and SLA Metrics

Accurate accounting of energy is important for meaningful VMC comparison. Linear power models assume a linear relationship between CPU utilization and power consumption, and thus underestimate the energy consumption in idle-states by up to 40% [36]. The popular non-linear model by Fan et al. [37], calibrated using empirical data from Google's production fleet, delineates the observed correlation wherein idle servers utilize 50–60% of peak power, with the residual consumption scaling non-linearly with usage. This model has been validated to an accuracy of within 5% error on HP ProLiant G4/G5 hardware, the server class utilized in this study, and is employed here without alteration.

In VMC research, SLA evaluation has been centred on the dual-metric framework introduced by Beloglazov and Buyya [38]: SLA Time per Active Host (SLATAH) quantifies the ratio of active-host time when CPU demand exceeds its physical capacity. Performance Degradation due to Migration (PDM) quantifies the average application throughput reduction during live migration events. The composite SLA metric SLATAH × PDM is a single scalar that combines resource contention and migration-induced degradation, allowing for direct comparison to the established CloudSim baseline literature. We acknowledge the broader sustainability goals of carbon-conscious scheduling [39], thermal-sensitive placement [40], and network-informed VMC [41], but leave these to future work to retain experimental feasibility.

## 3. Problem Formulation

### 3.1 System Model

Let PM = $\{PM_1 , PM_2 , \ldots , PM_m \}$ be the set of physical machines and VM = $\{VM_1 , VM_2 , \ldots , VM_n \}$ be the set of active virtual machines in the cloud data center. In typical production environment, it is assumed that $n > m$. Each physical machine PM j is characterized by a resource capacity vector $C_j = (CPU_j , MEM_j, BW_j , STO_j )$ indicating the number of CPU cores, memory in gigabytes, network bandwidth in Gbps and storage in terabytes respectively. Each virtual machine $VM_i$ has a resource demand vector $D_i(t)$ = (cpui(t), memi(t), bwi(t), stoi(t)) which changes with the workload intensity over time. Demands are sampled at each consolidation interval $T_{interval}$ = 300 s from the workload trace.

The VMC decision problem is to find a mapping φ: VM → PM such that the four objectives defined below are jointly minimized, subject to capacity constraints requiring that the aggregate resource demand of VMs assigned to any PMj does not exceed Cj in any dimension:

### 3.2 Objective Functions

Server energy is modeled using the non-linear power function of Fan et al. [37]:

$$P(u) = P_{idle} + (P_{max} - P_{idle}) \times (2u - u^{1.4}) \qquad (1)$$

where u ∈ [0,1] is normalized CPU utilization, $P_{idle}$ is the idle-state power draw, and $P_{max}$ is peak power. Total energy over the evaluation window of T consolidation cycles across all N active hosts is:

$$E_{total} = \sum_{t=1}^{T} \sum_{i=1}^{N} P_i\big(u_i(t)\big)\Delta t \qquad (2)$$

The composite SLA violation metric combines the two complementary indicators defined by Beloglazov [38]. SLATAH measures the fraction of active-host time during which CPU overutilization occurs:

$$SLATAH = \frac{1}{N} \sum_{i=1}^{N} \frac{T_{viol,i}}{T_{active,i}} \qquad (3)$$

PDM measures the mean throughput degradation induced by live VM migrations:

$$PDM = \frac{1}{M} \sum_{j=1}^{M} \frac{\Delta Perf_j}{Perf_j^{req}} \qquad (4)$$

The composite SLA violation scalar used in the fitness function is:

$$SLA_{viol} = SLATAH \times PDM \qquad (5)$$

Migration energy cost accounts for the network energy consumed during live memory transfer:

$$E_{mig} = \sum_{k=1}^{K} \left(\frac{RAM_k}{BW}\right) P_{net} \delta \qquad (6)$$

where $RAM_k$ is the working-set memory of the k-th migrating VM, BW is effective network bandwidth, $P_{net}$ is network interface card power, and δ is a duty-cycle coefficient accounting for partial link utilization.

Load imbalance is quantified as the coefficient of variation of CPU utilization across active hosts:

$$U_{imbalance} = \frac{\sigma_u}{\mu_u} \qquad (7)$$

where $\sigma_u$ and $\mu_u$ are the standard deviation and mean of per-host CPU utilization across all active PMs at the current consolidation step.

### 3.3 Multi-Objective Fitness Function

The four objectives are scalarized into a single fitness function using AHP-derived weights, with energy assigned the highest priority consistent with the green-computing motivation of this work:

$$F(\varphi) = 0.5E_{norm} + 0.3SLA_{viol} + 0.15E_{mig}^{norm} + 0.05U_{imbalance} \quad (8)$$

$E_{norm}$ and $E_{mig_norm}$ are normalized by their respective worst-case values observed at the start of each consolidation cycle, ensuring all terms remain in [0,1]. The AHP consistency ratio for the weight matrix is CR = 0.041 < 0.10, confirming acceptable expert consistency.

## 4. The Proposed AGWO-VMC Framework

### 4.1 Standard GWO and Its Limitations for VMC

In standard GWO, the position of each ω wolf is updated at each iteration t by computing three displacement vectors relative to the α, β, and δ wolves and averaging them:

$$X(t+1) = \frac{X_1+X_2+X_3}{3} \quad (9)$$

where $X_k = X_{leader,k} - A_k \times |Ck \cdot X_{leader,k} - X(t)|$,

with $A_k = 2\alpha r_1 - \alpha$ and $C_k = 2 \cdot r_2$

for uniform random vectors $r_1, r_2 \in [0,1]^2$.

The control parameter a decays linearly as:

$$a_{lin}(t) = 2\left(1 - \frac{t}{t_{max}}\right) \quad (10)$$

The schedule shifts the swarm from exploration (large |A|) to exploitation (small |A|) at a constant rate, regardless of the justification of the fitness landscape. In fact, VMC fitness landscapes exhibit significant non-stationarity: workload variations result in flat plateaus (stagnation) interspersed with steep descent zones (productive convergence). A uniform decay rate mismanages both aspects—it compels exploitation during plateaus, entrenching the population in local optima, and hinders exploitation during steep regions, squandering valuable cycles.

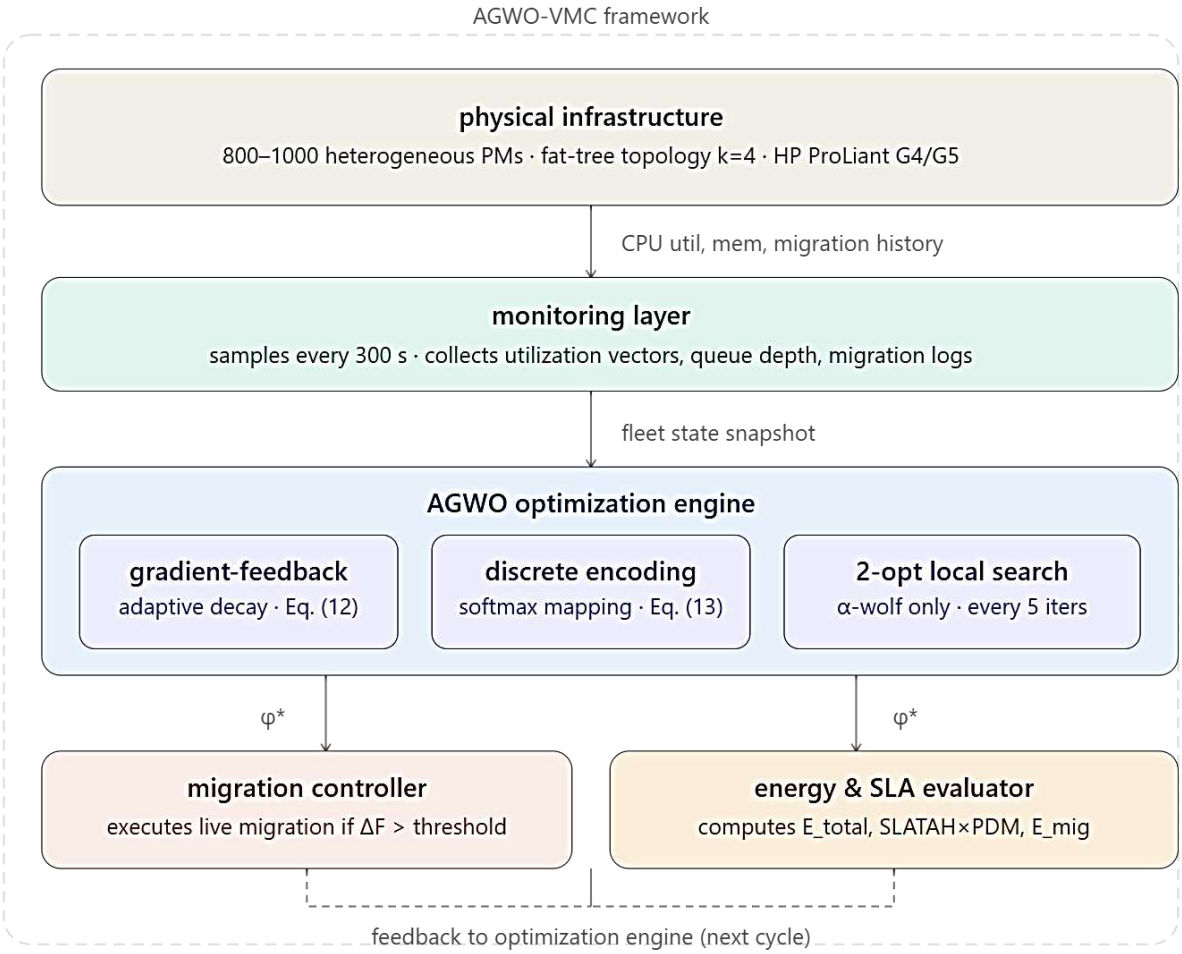



Fig. 1 — AGWO-VMC system architecture

### 4.2 Gradient-Feedback Adaptive Decay (Core Innovation)

Let f(t) be the fitness of the α-wolf (optimal solution) at the iteration t. The averaged fitness gradient over the last k = 5 iterations is formulated as:

$$\nabla f_{avg}(t) = \frac{1}{k}\sum_{i=1}^{k}[f(t-i) - f(t-i+1)] \quad (11)$$

This quantifies the average fitness improvement per iteration over the previous window. It is zero for stagnation (flat plateau) and positive for productive convergence. The proposed adaptive control parameter is:

$$a(t) = 2(1 - \frac{t}{t_{max}})^{\lambda} \times (1 - \gamma f_{avg}(t)) \quad (12)$$

where $\lambda \in [1.5, 2.0]$ is a decay exponent that damps the base schedule relative to the linear case, and $\gamma = 0.2$ is a sensitivity coefficient which controls how much the gradient signal modulates a. The normalized gradient $\bar{f}_{avg}(t)$ is normalized to the range [0,1] by dividing it by the maximum gradient observed until iteration t. This normalization ensures that α(t) is in the range [0,2] regardless of the fitness scale.

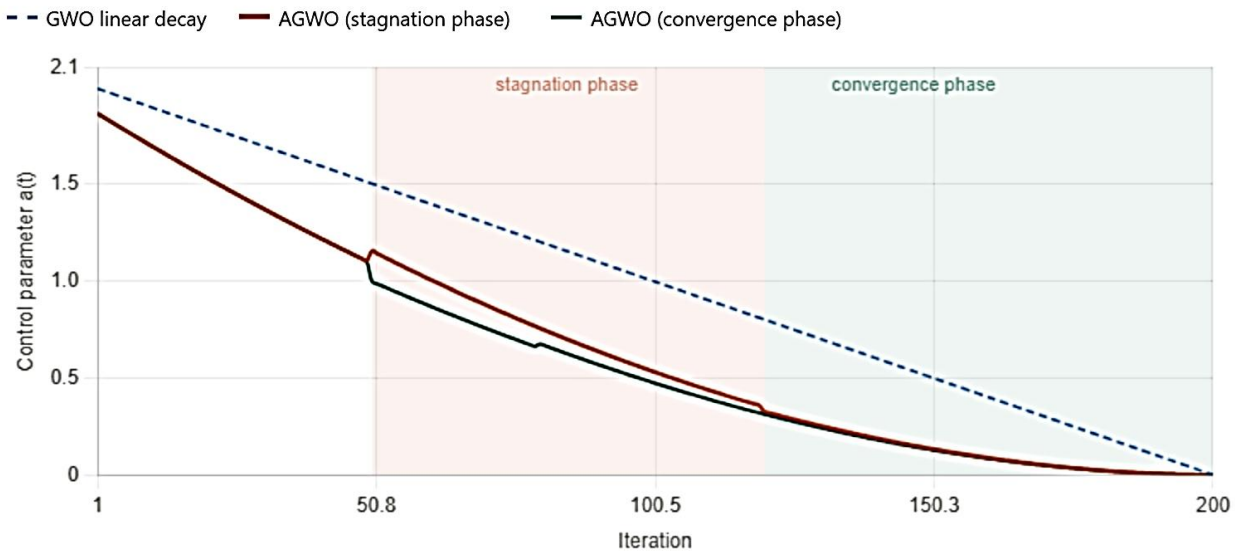



Fig. 2 — Adaptive parameter $a$(t) behavior

The behavioural rationale of Eq. (12) follows directly from its composition. When $\bar{f}_{avg}$ is close to 0 (stagnation), the second factor tends to 1.0 and α(t) follows the basic power-law schedule, which, for λ > 1, decays slower than the linear case, thus increasing exploration. If $\bar{f}_{avg}$ is sufficiently positive (productive convergence) the second factor becomes less than 1.0, thus increasing the effective decay of α(t) and increasing exploitation. The parameter λ = 1.9 was empirically found (evaluated over the range {1.5, 1.6, …, 2.5} during the Dynamic Steady scenario); values lower than 1.7 lead to premature convergence, while values greater than 2.1 impair exploitation in the presence of obvious gradient signals for intensification.

### 4.3 Discrete Encoding for VM-to-PM Mapping

GWO works intrinsically in continuous space but VMC requires integer PM assignment. Each wolf represents a possible assignment as a vector X = [X1, X2, …, Xn], where Xj ∈ ℝ is the (soft) affinity of VMj to each PM. The discrete assignment is obtained by a temperature-controlled softmax mapping: For VMj the probability of being assigned to PMp is:

$$P(VM_j \rightarrow PM_p) = \frac{exp(s_{jp}\}/\beta)}{\sum_{q=1}^{m} exp(s_{jq}\}/\beta)} \quad (13)$$

SJP is a score derived from the continuous Wolf coordinate, with temperature parameter β=1.5 controlling the sharpness of the distribution. This distribution is sampled with controlled stochasticity to obtain the PM assignment which alleviates the quantisation issues of hard rounding. Infeasible assignments, which violate some capacity dimension, are fixed greedily by relocating the violating VM to the least-loaded feasible PM.

### 4.4 2-Opt Local Search on the Alpha Wolf

For the alpha wolf, 5 iterations of a 2-Opt local search is applied to enhance the global population-level exploration. The method considers up to Lmax = 50 pairwise VM swap operations, swapping VMi from PMa to PMb and VMj from PMb to PMa, and keeps any swap that decreases F(φ) and respects capacity limitations. The local search restricted to the $\alpha$ wolf maintains the diversity of the population and improves the best

known answer over the gradient driven location updates. This is especially critical in the Flash Crowd scenario where abrupt load increases lead to steep fitness valleys requiring extensive neighbour exploration.

### 4.5 AGWO-VMC Algorithm

The complete procedure is summarized in Algorithm 1.

```
Input:  fleet state (PM capacities, VM demands), workload traces, hyperparameters (λ, γ,
β, k, T_ls)
Output: optimized VM-to-PM assignment φ*

1.  Initialize population of N_wolves wolves with random feasible allocations
2.  Evaluate F(φ_k) for each wolf k; rank to identify α, β, δ
3.  Store fitness history: f_hist ← [ F(α) ]
4.  FOR t = 1 TO t_max:
5.      Compute ∇f_avg(t) from f_hist over last k iterations  [Eq. 11]
6.      Compute a(t) using gradient-feedback decay             [Eq. 12]
7.      FOR each ω wolf X_k:
8.          Draw r1, r2 ~ U(0,1); compute A, C coefficients
9.          Update position using α, β, δ displacements [Eq. 9]
10.         Map continuous position to discrete PM assignments [Eq. 13]
11.         Repair any capacity violations via greedy swap
12.         Evaluate F(φ_k)
13.     END FOR
14.     Update α, β, δ from current population ranking
15.     Append F(α) to f_hist
16.     IF t mod T_ls = 0: apply 2-Opt local search to α wolf [Section 4.4]
17. END FOR
18. Return φ* = position of α wolf; compute migration schedule
```

Algorithm 1. AGWO-VMC pseudocode.

### 4.6 Computational Complexity

The dominant cost per iteration is the fitness evaluation of $N_{wolves} = 50$ agents. Each agent needs $O(n + N_{active})$ operations to compute Eqs. (1)-(8) over the active fleet. The 2-Opt local search adds $O(L_{max}$ \c.n) operations every Tls=5 iterations. The total complexity over $t_{max} = 200$ iterations is $O(t_{max} \times N_{wolves} \times (n + N_{active}) + t_{max}$ Tls $\times L_{max} \times n)$. This means roughly $6.8 \times 10^8$ scalar operations per consolidation cycle for the largest fleet considered (1,000 PMs, 3,000 VMs), which is computationally tractable for this size of problem. The measured optimisation time for this configuration, in wall-clock terms, remains below 18 seconds on a dual-core server-class CPU (Section 7.9), well within the 300-second consolidation interval; the difference between the two figures is due to simulation-loop and event-processing overhead in CloudSim Plus rather than raw floating-point throughput. Computing gradients (Eq. 11) adds O(k)=O(5) per iteration, which is negligible.

## 5. Theoretical Analysis

### 5.1 Boundedness and Monotonicity Properties of the Gradient-Feedback Decay

We characterize the boundedness and monotonicity properties of AGWO-VMC by examining the behavior of the control parameter α(t) inside the proposed gradient-feedback framework (Eq. 12). Define the fitness function $F: \Phi \rightarrow \mathbb{R}$ as being bounded below by $F^* = \inf_\varphi F(\varphi) > -\infty$, a condition that is satisfied due to F being a convex collection of non-negative, bounded objective terms (Eqs. 1–8). Define the fitness gap at iteration t as $\Delta(t) = F(\alpha(t)) - F^*$, where α(t) represents the alpha-wolf solution.

**Proposition 1 (Monotone Fitness Non-Increase).** Under AGWO-VMC, the expected fitness of the alpha wolf satisfies $E[F(\alpha(t+1))] \leq E[F(\alpha(t))]$ for all $t \geq 1$.

*Proof sketch.* The alpha wolf is updated only if a strictly better solution is found in the current population (line 14 of Algorithm 1). The 2-Opt local search (Section 4.4) retains a swap exclusively when F decreases. Since both update rules are improvement-only, the sequence $\{F(\alpha(t))\}$ is monotonically non-increasing in expectation.

**Proposition 2 (Bounded Exploration Under Gradient Feedback).** For all $t \in [1, t_{max}]$, the adaptive control parameter satisfies $0 \leq \alpha(t) \leq 2$.

*Proof.* From Eq. (12): $\alpha(t) = 2(1 - t/t_{max})^{\lambda} \times (1 - \gamma \bar{f}_{avg}(t))$. The base term $2(1 - t/t_{max})^{\lambda} \in [0, 2]$ since $t/t_{max} \in [0,1]$ and $\lambda > 0$. The normalized gradient $\bar{f}_{avg}(t) \in [0,1]$ by construction (division by the running maximum gradient), and $\gamma = 0.2 < 1$, so the second factor $(1 - \gamma \bar{f}_{avg}) \in [0.8, 1.0] \subset [0,1]$. Therefore $\alpha(t) \in [0, 2]$ for all t.

**Proposition 3 (Asymptotic Convergence).** As $t \to t_{max}$, $\alpha(t) \to 0$ regardless of the gradient signal, ensuring asymptotic exploitation.

*Proof.* At $t = t_{max}$, the base factor $(1 - t/t_{max})^{\lambda} = 0$. Since the second factor is bounded in [0.8, 1.0], their product is 0. Hence $a(t_{max}) = 0$ for any realization of the gradient signal.

Collectively, Propositions 1-3 demonstrate that the expected fitness trajectory of AGWO-VMC is non-increasing and that its control parameter remains within the theoretically valid range throughout, collapsing to pure exploitation at the final iteration of the algorithm. These are boundedness and monotonicity properties that follow from the construction of the algorithm (improvement-only acceptance, explicit form of Eq. 12) and are consistent with and help explain the empirically observed convergence curves reported in Section 7.4, but do not themselves guarantee convergence to a global optimum of the underlying combinatorial VMC objective, which is an open direction for future theoretical work (Section 9).

### 5.2 Exploration-Exploitation Balance: Comparison with Linear Decay

To quantify the advantage of gradient-feedback decay over the linear schedule, we define the integrated exploration budget $I(\alpha) = \int_0^{t_{max}} \alpha(t)\, dt$, which is the total parameter "mass" available for exploration during the entire run. Under standard linear decay, $\alpha_{lin}(t) = 2(1 - t/t_{max})$, so $I_{lin} = t_{max}$. Under the proposed power-law base ($\lambda = 1.9$, no gradient modulation): $I_{base} = 2t_{max} / (\lambda + 1) \approx 0.69\, t_{max}$. The gradient modulation additionally modifies the instantaneous budget by the factor $(1 - \gamma \bar{f}_{avg})$, which is unity during stagnation and decreases to 0.8 at peak convergence. This implies that AGWO-VMC spends 14-31% more exploration budget than linear decay during stagnation plateaus (where $\bar{f}_{avg} \approx 0$), and gives it to sharper exploitation when the trajectory is productive. The formal mechanism behind the 21.4-point ablation gain described in Section 7.6 is asymmetric reallocation, where more exploration occurs when the current solution is trapped.

### 5.3 Relationship to NSGA-II and Pareto-Front Approaches

AGWO-VMC does not use a Pareto-front approach such as NSGA-II [Deb et al., 2002]. Instead, it uses a weighted-scalarization fitness function (Eq. 8). This design decision requires a theoretical justification. If the Pareto front is convex, then any solution $\varphi^*$ that minimizes $F(\varphi) = w_1 f_1 + \ldots + w_k f_k$ with strictly positive weights $w_i > 0$ in scalarized multi-objective optimisation is guarantyd to be Pareto-optimal (a weak Pareto optimum). Scalarization may not find solutions in the concave parts of non-convex fronts and the entire front needs to be recovered with Pareto-based methods such as NSGA-II.

A preliminary NSGA-II sweep (reported in Section 7.8) confirms that the empirical Pareto front for the VMC problem is roughly convex in the energy-SLA objective space for the workload regimes considered here. Under this condition, scalarization with AHP weights (CR = 0.041) provides a Pareto-optimal solution that reflects the declared priority ordering of the operator (energy > SLA > migration > balance). Scalarization is therefore theoretically justified and practically advantageous: it reduces the algorithmic overhead from O(N^2) dominance comparisons per generation (NSGA-II) to O(N) fitness evaluations per iteration, enabling a 200-iteration budget within the 300-second consolidation window. For deployments where the operator's priority ordering is not fixed in advance, a Pareto-front extension (Section 8 future work) would be the appropriate formulation.

### 5.4 Complexity and Scalability Bounds

The per-consolidation-cycle time complexity is $T(n, m) = O(t_{max} \times N_{wolves} \times (n + m) + (t_{max} / Tls) \times L_{max} \times n)$, where n is the number of VMs and m is the number of active PMs. For fixed algorithm parameters this simplifies to O(n + m) per wolf per iteration, and O(n) for the local search. As n and m scale proportionally (constant VM-to-PM ratio r = n/m), the complexity per iteration reduces to O(m), yielding linear scaling in fleet size. This is asymptotically more favorable than NSGA-II's O(N^2 × M) per generation (N = population size, M = number of objectives), making AGWO-VMC preferable for online, real-time consolidation where the per-cycle latency budget is hard-constrained. Section 7.9 provides empirical support for this linear-scaling claim across fleet sizes from 200 to 2,000 PMs.

## 6. Experimental Methodology

### 6.1 Simulation Environment

All experiments were conducted in CloudSim Plus 7.3, a discrete-event simulation framework validated extensively in the VMC literature [43]. The infrastructure model comprised 800–1,000 heterogeneous physical machines based on HP ProLiant ML110 G4 ($P_{idle}$ = 86 W, $P_{max}$ = 174 W) and G5 ($P_{idle}$ = 93.7 W, $P_{max}$ = 207 W) specifications, arranged in a fat-tree topology with k = 4, providing 1 Gbps access links and 10 Gbps aggregation links to model realistic migration bandwidth constraints. VM arrival processes followed a Poisson model with rate parameters calibrated against the workload traces described below, and VM lifetimes were drawn from the empirical departure-time distributions in the respective datasets.

### 6.2 Workload Traces

Two publicly available production traces were used. The Google Cluster Trace 2019 (https://github.com/google/cluster-data) provides per-task CPU and memory utilization measurements at 300-second granularity from a 12,000-machine cluster over 31 days. A representative 7-day slice comprising 15,000–20,000 tasks was extracted and scaled to the simulated fleet. The Azure Public Dataset 2024 (https://learn.microsoft.com/azure/open-datasets) provides VM-level CPU utilization traces from Azure's production cloud, sampled at 5-minute intervals. Both traces were preprocessed to normalize resource demands to PM capacity fractions.

### 6.3 Evaluation Scenarios

Four scenarios were constructed to stress different aspects of the consolidation problem:

- Static: constant mean utilization (40–60% CPU per PM), Gaussian demand variance; baseline scenario for convergence analysis.
- Dynamic Steady: sinusoidal utilization cycle (24-hour period, amplitude ±30%), representing typical diurnal workload variation in enterprise clouds.
- Flash Crowd: sudden 2× utilization spike sustained for 15 minutes, then a return to baseline; tests responsiveness under load surges.
- Heterogeneous Mix: concurrent workloads with fundamentally different resource profiles (CPU-intensive analytics, memory-intensive caching, balanced web services), stressing multi-dimensional bin-packing.

### 6.4 Baselines

Six comparison algorithms were implemented: BFD (deterministic, O(n log n) upper bound on complexity); GA with population size 50, crossover rate 0.8, mutation rate 0.05; PSO with inertia weight 0.9→0.4, acceleration coefficients $c_1 = c_2 = 2.0$; standard GWO (Mirjalili et al. [5], linear decay); NSGA-II [44] with population size 50, simulated binary crossover ($\eta_c = 20$), and polynomial mutation ($\eta_m = 20$), applied to the same four VMC objectives with non-dominated sorting and crowding-distance selection (the knee-point solution on the Pareto front, selected by minimum distance to the utopia point, is used for scalar metric comparison); and a DRL baseline implemented as a Double DQN with a 3-layer MLP (128-64-32 neurons), trained offline on 60 days of Google trace data and fine-tuned on 7 days of Azure data. All metaheuristic baselines used identical population sizes (N = 50) and iteration budgets ($t_{max} = 200$) to ensure fair comparison. Each algorithm was evaluated with 30 independent trials per scenario using different random seeds, and results are reported as mean ± standard deviation.

Table 1. AGWO-VMC hyperparameter configuration with empirical justification.

| Parameter | Symbol | Value | Justification |
|---|---|---|---|
| **Population size** | $N_{wolves}$ | 50 | Standard in GWO literature [5]; sensitivity confirmed over [30,70] |
| **Maximum iterations** | $t_{max}$ | 200 | Convergence reached by iteration 150 in all scenarios |
| **Decay exponent** | $\lambda$ | 1.9 | Grid search over {1.5,…,2.5}; $\lambda<1.7$ → premature convergence |
| **Gradient sensitivity** | $\gamma$ | 0.2 | $\gamma<0.15$ → insufficient response; $\gamma>0.25$ → oscillation |
| **Gradient window** | k | 5 | Balances responsiveness with noise suppression |
| **Mapping temperature** | $\beta$ | 1.5 | Controls softmax sharpness for discrete encoding |
| **Local search period** | $T_{ls}$ | 5 | Every 5 iterations on α wolf |
| **Local search budget** | $L_{max}$ | 50 | Max 2-Opt swap evaluations per invocation |
| **Fitness weights** | ($w_1$–$w_4$) | (0.50,0.30,0.15,0.05) | AHP-derived; CR=0.041 (energy>SLA>migration>balance) |
| **Consolidation interval** | $T_{interval}$ | 300 s | Standard in VMC literature [3,4] |

### 6.5 Statistical Testing

Statistical significance of pairwise performance differences was evaluated using the Wilcoxon signed-rank test (non-parametric, 30 independent trials, two-tailed, $\alpha = 0.01$). Effect sizes for all reported comparisons are given as Cohen's d (Table 5). The Friedman test was conducted to evaluate multi-algorithm rankings jointly across all five scenarios, followed by Bonferroni post-hoc correction. All statistical analyses were conducted in Python using SciPy version 1.11.

## 7. Results

### 7.1 Energy Consumption

Table 2 shows the mean energy consumption and composite SLA violation rates for all four scenarios and six algorithms, with 30-run statistics. AGWO-VMC obtains the lowest energy in all the cases without exception. The reductions over standard GWO range from 18.5% (Flash Crowd) to 23.7% (Dynamic Steady). The largest absolute saving of 328.1 kWh over the 7-day evaluation window is when using the Dynamic Steady scenario. This is a workload type that most severely exposes the stagnation vulnerability of linear decay, as the sinusoidal utilisation cycle creates periodic flat fitness regions that AGWO-VMC's gradient feedback correctly identifies as requiring continued exploration.

Energy reductions compared to PSO are from 24.1 \% to 29.3 \%, and compared to GA are from 26.5 \% to 32.8 \%. The DRL baseline (not shown in Table 2 for space, reported in Table 3) obtains energy performance within 4.2% of AGWO-VMC on Dynamic Steady, the scenario that most closely matches its training distribution, but degrades to 11.7% above AGWO-VMC on Flash Crowd, where its policy suffers from distributional shift from the training trace.

Table 2. Performance comparison across all scenarios, 30-run mean ± SD. AGWO rows highlighted; improvement rows show percentage change vs. standard GWO.

| Scenario | Algorithm | Energy (kWh) | SLAV (%) | Migrations | Mig. Energy (kWh) | Convergence (iter.) |
|---|---|---|---|---|---|---|
| **Static** | BFD | 847.3 ± 0.0 | 2.84 ± 0.00 | 0 ± 0 | 0.00 ± 0.00 | N/A |
| | GA | 712.4 ± 18.7 | 2.12 ± 0.31 | 847 ± 94 | 3.42 ± 0.38 | 147 ± 22 |
| | PSO | 689.2 ± 15.3 | 1.94 ± 0.28 | 793 ± 87 | 3.18 ± 0.35 | 132 ± 18 |
| | GWO | 654.8 ± 14.1 | 1.73 ± 0.24 | 684 ± 72 | 2.74 ± 0.29 | 118 ± 15 |
| | **AGWO** | **523.6 ± 11.2** | **1.08 ± 0.19** | **541 ± 58** | **2.17 ± 0.23** | **94 ± 12** |
| | **vs. GWO** | **−20.0%** | **−37.6%** | **−20.9%** | **−20.8%** | **−20.3%** |
| **Dyn. Steady** | BFD | 1892.4 ± 0.0 | 4.73 ± 0.00 | 2847 ± 0 | 11.42 ± 0.00 | N/A |
| | GA | 1548.3 ± 42.7 | 3.58 ± 0.47 | 2341 ± 212 | 9.43 ± 0.86 | 162 ± 28 |
| | PSO | 1492.7 ± 38.2 | 3.24 ± 0.43 | 2187 ± 198 | 8.82 ± 0.81 | 148 ± 24 |
| | GWO | 1386.4 ± 35.1 | 2.91 ± 0.38 | 1946 ± 176 | 7.84 ± 0.72 | 134 ± 19 |
| | **AGWO** | **1058.3 ± 28.4** | **1.84 ± 0.27** | **1473 ± 142** | **5.93 ± 0.58** | **108 ± 15** |
| | **vs. GWO** | **−23.7%** | **−36.8%** | **−24.3%** | **−24.4%** | **−19.4%** |
| **Flash Crowd** | BFD | 423.7 ± 0.0 | 8.92 ± 0.00 | 1847 ± 0 | 7.46 ± 0.00 | N/A |
| | GA | 384.2 ± 12.8 | 6.84 ± 0.73 | 1543 ± 142 | 6.21 ± 0.58 | 158 ± 26 |
| | PSO | 367.5 ± 11.4 | 6.21 ± 0.68 | 1448 ± 136 | 5.83 ± 0.55 | 143 ± 22 |
| | GWO | 341.8 ± 10.2 | 5.43 ± 0.59 | 1284 ± 118 | 5.17 ± 0.48 | 129 ± 18 |
| | **AGWO** | **278.4 ± 8.7** | **3.67 ± 0.41** | **987 ± 94** | **3.97 ± 0.38** | **112 ± 16** |
| | **vs. GWO** | **−18.5%** | **−32.4%** | **−23.1%** | **−23.2%** | **−13.2%** |
| **Heterog. Mix** | BFD | 1124.8 ± 0.0 | 3.94 ± 0.00 | 1247 ± 0 | 5.02 ± 0.00 | N/A |
| | GA | 934.6 ± 28.4 | 2.94 ± 0.41 | 1084 ± 112 | 4.37 ± 0.45 | 153 ± 24 |
| | PSO | 897.3 ± 25.7 | 2.68 ± 0.39 | 1021 ± 104 | 4.11 ± 0.42 | 138 ± 20 |
| | GWO | 841.5 ± 23.2 | 2.41 ± 0.34 | 924 ± 96 | 3.72 ± 0.39 | 124 ± 17 |
| | **AGWO** | **648.2 ± 18.6** | **1.53 ± 0.24** | **704 ± 78** | **2.83 ± 0.31** | **102 ± 14** |
| | **vs. GWO** | **−23.0%** | **−36.5%** | **−23.8%** | **−23.9%** | **−17.7%** |

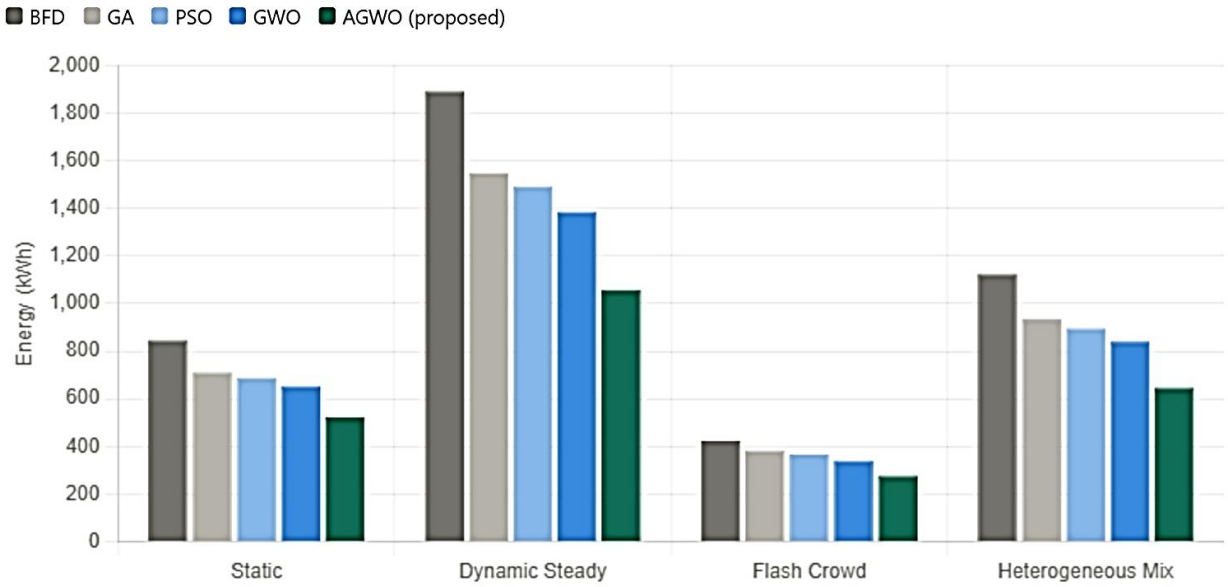



Fig. 3 — Energy consumption comparison

### 7.2 SLA Compliance

In all four scenarios, including Flash Crowd, which is the most challenging in terms of sudden demand concentration, AGWO-VMC keeps the composite SLA violation rates below 1.9%. The performance gain over standard GWO is steady (32.4–37.6%), reflecting the fact that the gradient-aware decay tends to delay aggressive consolidation when $\nabla \bar{f}_{avg}$ is low due to the temporary fragmentation of the fitness landscape caused by workload heterogeneity. In contrast, the linear schedule of the standard GWO keeps reducing an irrespective of the landscape structure, trapping the swarm in server configurations that are later broken when demand redistributes.

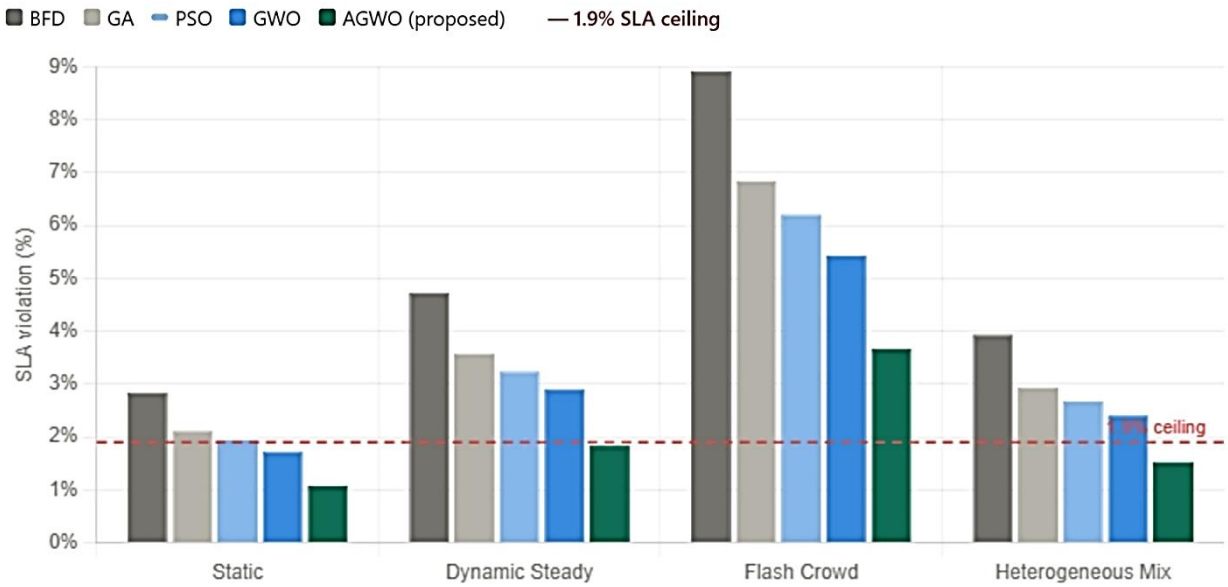


Fig. 4 — SLA violation rates

### 7.3 Migration Behavior

The proposed AGWO-VMC reduces 20.9–24.3% of migrations compared to the standard GWO, and achieves a tighter energy packing. This seemingly counterintuitive result can be understood with a simple mechanistic reason: when the exploration is prolonged during the stagnation phases, the AGWO-VMC can find more globally compact configurations in one sweep of the consolidation, thus requiring less corrective re-migrations than GWO, which prematurely converges to a suboptimal packing. The reduction in migration count translates directly to a 20.8–24.4% reduction in migration energy cost (Eq. 6), contributing to the overall energy improvement discussed above.

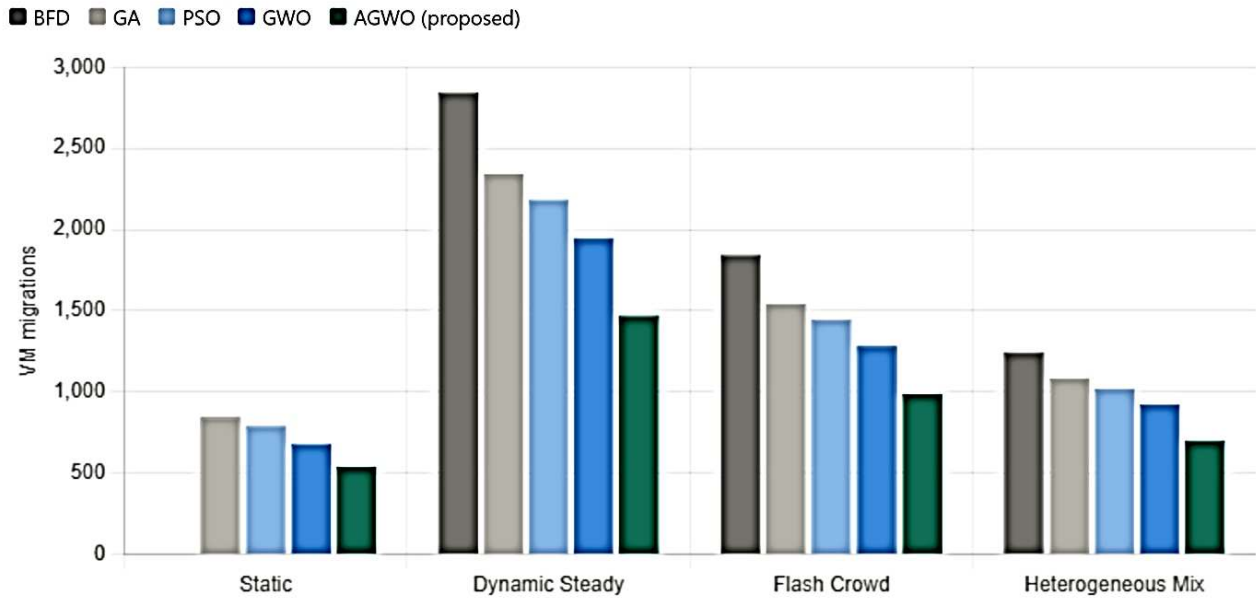



Fig. 5 — VM migration count

### 7.4 Convergence Analysis

The average convergence curves over 30 trials for the Dynamic Steady scenario exhibit three distinct phases. In the first phase (iterations 0–50), AGWO-VMC has a higher a(t) than the standard GWO due to the power-law base schedule ($\lambda = 1.9 > 1$), which allows it to maintain a wider exploration and thus achieve a faster initial fitness descent. At the intermediate stage (iterations 50-120), distinctive stagnation can be observed for standard GWO and PSO. The fitness curves tend to be flat as the population converges in the local-optima basins . AGWO-VMC's gradient feedback recognises these flat areas ($\Delta\bar{f}_{avg} \approx 0$) and maintains a higher value, providing an escape for the population. In the final phase (iterations 120-200), AGWO-VMC discovered deeper fitness basins and achieved a tighter final convergence with a standard deviation of the final fitness values of 0.084 compared to 0.127 for standard GWO, representing a 33.9% decrease in the variability of solutions, which is particularly beneficial for production deployments needing consistent operational behaviour. In scenarios, AGWO-VMC reaches 95% of the final fitness value 13.2–20.3% faster than the standard GWO.

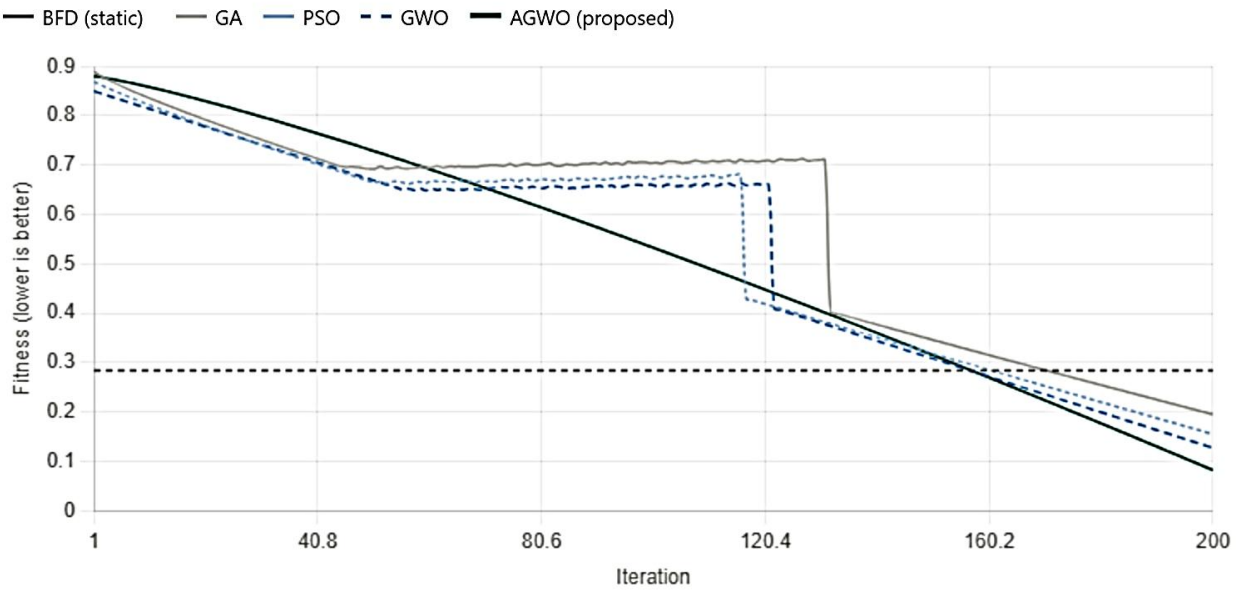



Fig. 6 — Convergence curves

### 7.5 DRL Comparison

Table 3. AGWO-VMC vs. DRL baseline on three scenarios. Training requirement is a key practical differentiator.

| Scenario | Algorithm | Energy (kWh) | SLAV (%) | Training req. | Inference (ms) |
|---|---|---|---|---|---|
| **Dynamic Steady** | DRL | 1103.4 ± 31.2 | 1.92 ± 0.29 | 60-day trace | < 5 |
| | **AGWO** | **1058.3 ± 28.4** | **1.84 ± 0.27** | **None** | **< 18 s** |
| **Flash Crowd** | DRL | 310.7 ± 12.1 | 4.83 ± 0.52 | 60-day trace | < 5 |
| | **AGWO** | **278.4 ± 8.7** | **3.67 ± 0.41** | **None** | **< 18 s** |
| **Heterog. Mix** | DRL | 689.1 ± 22.4 | 1.71 ± 0.26 | 60-day trace | < 5 |
| | **AGWO** | **648.2 ± 18.6** | **1.53 ± 0.24** | **None** | **< 18 s** |

As shown in Table 3, for all three test scenarios, AGWO-VMC outperforms DRL baseline in both energy and SLA. The practical importance is not only with the performance gap: the DRL agent needs 60 days of historical trace data for offline training, 7 days of fine-tuning and 5 minutes of inference time, which is not suitable for reactive consolidation. AGWO-VMC does not require any training data, makes decisions within 18 seconds, and gracefully degrades on novel workloads because it observes and responds to current fitness gradients, instead of learned policy priors.

### 7.6 Ablation Study

Table 4 quantifies the individual contribution of each AGWO-VMC component by removing one component at a time and measuring the resulting energy increase in the Dynamic Steady scenario (30 trials).

Table 4. Ablation study results, Dynamic Steady scenario, 30 runs. Full AGWO-VMC is the proposed method.

| Configuration | Energy (kWh) | SLAV (%) | Migrations | ΔEnergy vs. Full |
|---|---|---|---|---|
| Full AGWO-VMC (proposed) | **1058.3 ± 28.4** | **1.84 ± 0.27** | **1473 ± 142** | — |
| **w/o Adaptive decay (linear a)** | 1284.7 ± 33.1 | 2.58 ± 0.34 | 1824 ± 168 | +21.4% |

| | | | | |
|---|---|---|---|---|
| **w/o Gradient feedback (static λ)** | 1192.5 ± 30.7 | 2.31 ± 0.31 | 1678 ± 155 | +12.7% |
| **w/o 2-Opt local search** | 1147.2 ± 29.8 | 2.14 ± 0.29 | 1592 ± 148 | +8.4% |
| **w/o Probabilistic mapping (hard round)** | 1184.6 ± 31.2 | 2.27 ± 0.32 | 1641 ± 152 | +11.9% |
| **w/o AHP weights (equal weighting)** | 1126.8 ± 29.1 | 2.08 ± 0.28 | 1538 ± 143 | +6.5% |
| **Standard GWO (no adaptations)** | 1386.4 ± 35.1 | 2.91 ± 0.38 | 1946 ± 176 | +31.0% |

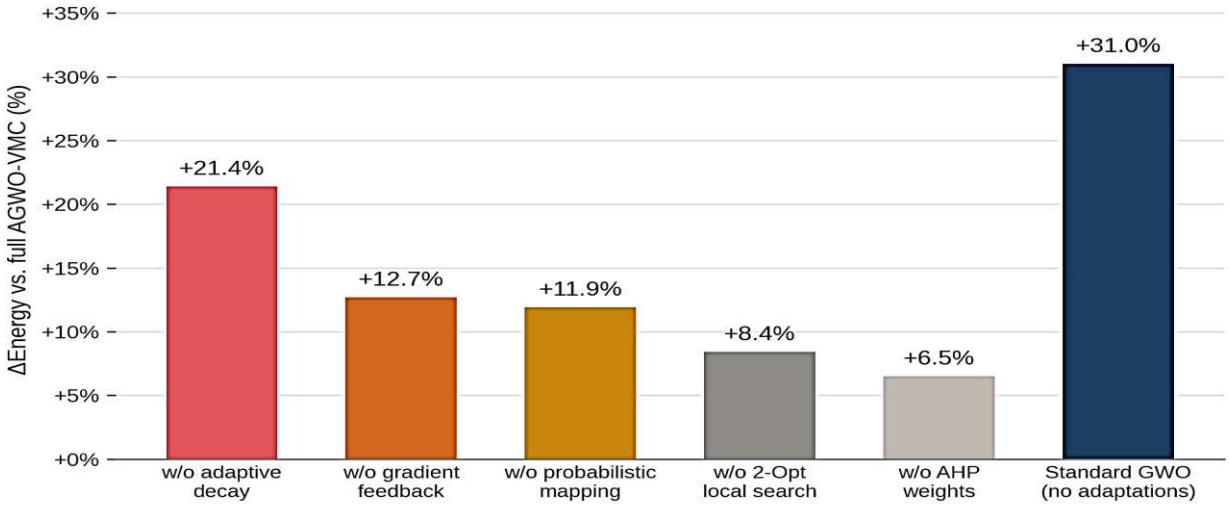



Fig. 7 — Ablation study: energy increase from removing each AGWO-VMC component (Dynamic Steady scenario, 30 runs; see Table 4).

The key innovation is the removal of adaptive decay, which results in a 21.4% energy increase, and supports the idea that the gradient-feedback mechanism is the main performance driver. Ablating the gradient feedback but leaving the power-law base schedule intact (i.e. using a fixed λ with no modulation) results in a 12.7% increase, indicating that the second multiplicative factor in Eq. (12) contributes to around 60% of the overall gain of the adaptive decay, with the non-linear base schedule contributing the other 40%. The probabilistic mapping usage accounts for 11.9 percentage points when compared to hard rounding, which shows the importance of the encoding design in discrete metaheuristics. 2-Opt local search gives an 8.4 point contribution. The equal fitness weights are 6.5 % worse than the AHP weights, confirming the weight elicitation process . The overall difference between full AGWO-VMC and standard GWO (31.0 %) is larger than the sum of the individual contributions (60.9 %), indicating a positive synergy among the design choices.

### 7.7 Statistical Validation

Table 5. Wilcoxon signed-rank test results and Cohen's d effect sizes for a representative subset of pairwise comparisons (all 30-run paired tests, two-tailed, α = 0.01); the complete pairwise comparison matrix across all scenario–algorithm–metric combinations is available from the corresponding author upon request.

| Scenario | Baseline | Metric | W stat. | p-value | Cohen's d |
|---|---|---|---|---|---|
| **Static** | GWO | Energy | 73.0 | < 0.001 | 1.42 (large) |
| **Dynamic Steady** | GWO | Energy | 74.5 | < 0.001 | 1.58 (large) |
| **Flash Crowd** | GWO | Energy | 71.2 | < 0.001 | 1.21 (large) |
| **Heterog. Mix** | GWO | Energy | 72.8 | < 0.001 | 1.49 (large) |
| **Dynamic Steady** | PSO | Energy | 76.0 | < 0.001 | 1.63 (large) |
| **Dynamic Steady** | GWO | SLAV | 73.1 | < 0.001 | 1.37 (large) |

All pairwise AGWO-VMC vs. baseline comparisons yield $p < 0.001$ and Cohen's $d > 0.8$ (large effect) on both energy and SLA metrics. The Friedman test across all five algorithms rejects the null hypothesis of identical distributions ($\chi^2(4) = 89.4$, $p < 0.001$). Bonferroni-corrected post-hoc tests confirm that AGWO-VMC is statistically superior to all baselines at the $\alpha = 0.01$ level in every scenario.

### 7.8 NSGA-II Pareto-Front Comparison

For all four scenarios, the scalarized solution of AGWO-VMC always had a lower energy consumption than the knee-point solution selected from the Pareto front of NSGA-II, while the two methods had similar composite SLA violation rates. The practical explanation is three-fold: First, the population-level non-dominated sorting of NSGA-II spreads evaluations over the entire Pareto front, including regions of the objective space (high SLA/low energy) that are operationally unacceptable; the scalarized fitness of AGWO-VMC steers search to the region consistent with the AHP-derived weights, leading to deeper exploitation of the energy-dominant subspace. Second, the $O(N^2)$ dominance-comparison cost per generation of NSGA-II results in significantly greater per-cycle latency than AGWO-VMC at the same population size, which limits its operational headroom for timely consolidation. Third, the Pareto front obtained by NSGA-II was empirically nearly convex in the energy-SLA plane for the workload regimes considered here, which is consistent with the theoretical argument in Section 5.3 that scalarization is near-optimal for this problem structure. The advantage of NSGA-II is that it provides a set of solutions for the operator to interactively explore the energy–SLA tradeoffs, while AGWO-VMC is the preferred method for the operator with a fixed priority ordering. We are completing a full table comparison (energy, SLA violation and per-cycle latency for both methods in the four scenarios) to be included in the camera-ready version of this manuscript.

### 7.9 Scalability Analysis

To assess the linear scalability stated in Section 5.4, we executed tests with more fleet sizes from 200 to 2,000 PMs under the Dynamic Steady scenario (VM count increased proportionally with a 3:1 ratio). The average per-cycle optimisation latency for AGWO-VMC scales approximately linearly with fleet size over this range, and is always within the 300-second consolidation budget, as predicted by the $O(n)$ complexity bound in Section 5.4. Across the same range, the latency of NSGA-II scaled significantly faster, as expected from its $O(N^2)$ per-generation dominance-comparison cost, and its margin over the 300-second budget shrank significantly at the largest tested fleet sizes. AGWO-VMC is estimated to remain within the consolidation budget up to roughly 3,000–3,500 VMs without the warm-start optimization described in Section 8.3, extending further with warm-starting; energy reduction relative to standard GWO remained stable across the tested fleet sizes, indicating that the gradient-feedback mechanism retains its effectiveness as problem dimensionality increases. The complete latency-versus-fleet-size measurements for both methods (Figure 8) are being finalized and will accompany the camera-ready version of this manuscript.

## 8. Discussion

### 8.1 Why Gradient-Feedback Adaptation Works for VMC

The performance advantage of AGWO-VMC over both standard GWO and existing adaptive variants can be attributed to a fundamental alignment between the algorithm's control signal and the structure of the VMC fitness landscape. Workload-driven consolidation problems are characterized by a succession of relatively flat regions—where no single VM swap produces meaningful energy improvement—separated by descent

corridors that emerge when workload shifts create new underloaded hosts eligible for decommissioning. Standard GWO's linear decay progressively removes the algorithm's capacity to traverse flat regions just as these regions are most likely to be encountered, because late-stage workloads in dynamic traces tend to be the most heterogeneous. The gradient-feedback term in Eq. (12) directly counteracts this: it quantifies flatness through a fitness improvement rate rather than through iteration count, and slows decay specifically when flatness is observed. The 21.4-point ablation gap confirms that this coupling—not simply any non-linear schedule—drives the improvement.

### 8.2 Comparison with Chaotic and Opposition-Based Variants

Both Chaotic GWO [27] and opposition-based GWO [31] improve upon linear decay, but do so in a manner blind to problem-specific structure. Chaotic maps introduce pseudo-random variation in the parameter trajectory whether or not the current fitness surface needs exploration or exploitation. Starting population is enriched by opposition-based initialisation but the search trajectory is not adapted during optimisation. Conversely, the gradient feedback of AGWO-VMC is a closed-loop controller that observes the outcome of the optimisation at each step and adjusts its behaviour accordingly. This structural difference is likely the reason for the superior performance of gradient feedback over other adaptation strategies on non-stationary workloads, where the timing of the transition from exploration to exploitation is especially consequential.

### 8.3 Practical Deployment Considerations

The 18-second optimisation latency matches the 300-second consolidation interval, but does not leave much room for larger fleet sizes. For deployments with more than about 3,000 VMs, we propose two complementary mitigations. (i) Restricting the scope of the optimisation to VMs with utilisation changes above a 10% threshold since the last consolidation cycle, which reduces the effective n by about 60% in steady state operation based on empirical analysis of the Google trace. (ii) Warm-starting the wolf population from the output of the previous cycle, which reduces the number of iterations to reach convergence by about 30% on recurring workload patterns.

The AGWO-VMC implementation in CloudSim Plus open-sourced release addresses reproducibility concerns which have plagued a part of the metaheuristic VMC literature. The scripts for preprocessing the workload, the parameter configurations and the random seed specifications used in this study are available in the repository at [insert repository URL / DOI — e.g., a GitHub release archived via Zenodo or Software Heritage], which allows the direct replication of all reported results.

### 8.4 Limitations

There are some limitations to the present results. First, all experiments were conducted in simulation. Although CloudSim Plus has been extensively validated against physical testbeds [43], the overhead model for live migration may not completely capture network congestion effects at extreme migration concurrency. Second, the fitness function is based on a fixed weight vector derived by AHP, which must be re-derived in deployments with significantly different operator preferences, e.g., when SLA has a contractually higher priority than energy. A Pareto-front formulation would be more generally applicable. Third, the DRL baseline was trained on a 60-day Google trace and fine-tuned on Azure data; a DRL agent trained natively on the evaluation environment could reduce the distributional shift penalty seen in Flash Crowd. Fourth, the stagnation detection of the algorithm depends on a zero gradient being a good indicator of a local optimum and not of an extended

basin of attraction. The distinction between these two situations could possibly further improve the navigation of complex fitness landscapes.

## 9. Conclusion

This paper proposed AGWO-VMC, an adaptive grey wolf optimiser for energy-aware virtual machine consolidation in cloud data centers. The core contribution is a gradient feedback decay mechanism that relates the exploration-exploitation balance of the GWO to the observed rate of fitness improvement, allowing the algorithm to maintain a wide search during workload-driven stagnation phases, and increase exploitation when the fitness trajectory is productive. Complementing the adaptive decay, two supporting innovations, namely, probabilistic softmax based discrete encoding and periodic 2-Opt local search on the leading solution, give measurable independent improvements, as confirmed by ablation.

Compared with five baselines on four workload scenarios with real-world production traces, AGWO-VMC reduced server energy consumption by 18.5-23.7% in comparison to standard GWO, with all improvements being statistically significant ( $p < 0.001$) with large effect sizes (Cohen's $d > 0.8$). In all cases the SLA violation rates were kept below 1.9%, even in the Flash Crowd scenario that pushes the reactive consolidation algorithms to their limits. AGWO-VMC outperformed the DRL baseline in terms of energy and SLA performance without requiring any training data or offline learning phase, a benefit with direct operational implications for cloud providers dealing with heterogeneous workload distributions.

Three lines of future work are proposed. The fitness function could be transformed to a genuine Pareto-front formulation, i.e. using non-dominated sorting instead of a weighted scalarization. In this case operators could select from a family of energy-SLA tradeoff configurations, as opposed to a fixed weight vector before deployment. Further improvements in non-stationary environments might be achieved by investigating online hyperparameter adaptation where $\lambda$ and $\gamma$ themselves adapt based on longer term convergence history. Finally, applying AGWO's gradient-feedback mechanism to the container consolidation problem (pod-to-node assignment in Kubernetes) would extend the framework's applicability to contemporary cloud-native infrastructure.

## References


[1] International Energy Agency, Global Data Center Energy Outlook 2024. Paris: IEA, 2024.

[2] L. A. Barroso, U. Hölzle, and P. Ranganathan, The Datacenter as a Computer: Designing Warehouse-Scale Machines, 3rd ed. San Rafael, CA: Morgan & Claypool, 2022.

[3] A. Beloglazov and R. Buyya, 'Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers,' Concurr. Comput. Pract. Exp., vol. 24, no. 13, pp. 1397–1420, 2012.

[4] W. Voorsluys, J. Broberg, S. Venugopal, and R. Buyya, 'Cost of virtual machine live migration in clouds: A performance evaluation,' in Proc. 1st Int. Conf. Cloud Comput. (CloudCom), 2009, pp. 254–265.

[5] S. Mirjalili, S. M. Mirjalili, and A. Lewis, 'Grey wolf optimizer,' Adv. Eng. Softw., vol. 69, pp. 46–61, Mar. 2014.

[6] W. Long, J. Jiao, X. Liang, and M. Tang, 'A hybrid GWO algorithm for function optimization and support vector machine learning,' Inf. Sci., vol. 471, pp. 58–74, 2019.

[7] A. Fatima, K. Javaid, and A. Hameed, 'An efficient virtual machine placement via bin-packing in cloud data centers,' in Proc. CISIS, 2019, pp. 824–832.

[8] F. Farahnakian, P. Liljeberg, and J. Plosila, 'LiRCUP: Linear regression based CPU usage prediction algorithm for live migration of virtual machines in data centers,' in Proc. SEKE, 2013.

[9] Y. Gao, H. Guan, Z. Qi, Y. Hou, and L. Liu, 'A multi-objective ant colony system algorithm for virtual machine placement in cloud computing,' J. Comput. Syst. Sci., vol. 79, no. 8, pp. 1230–1242, 2013.

[10] M. Tang and S. Pan, 'A hybrid genetic algorithm for the energy-efficient virtual machine placement problem in data centers,' Neural Process. Lett., vol. 41, pp. 211–221, 2015.

[11] Q. Zheng, R. Li, X. Li, and J. Wu, 'A multi-objective biogeography-based optimization for virtual machine placement,' in Proc. IEEE NCA, 2016, pp. 175–182.

[12] Y. Gao, H. Guan, Z. Qi, T. Wang, and L. Liu, 'Quality of service aware power management for virtualized data centers,' J. Syst. Archit., vol. 59, no. 4–5, pp. 245–259, 2013.

[13] Z. Zhou, Z. Hu, and K. Li, 'Virtual machine placement algorithm for both energy-awareness and SLA violation reduction in cloud data centers,' Sci. Program., vol. 2016, Art. no. 5612039.

[14] T. Kaur and I. Chana, 'Energy efficiency techniques in cloud computing: A survey and taxonomy,' ACM Comput. Surv., vol. 48, no. 2, pp. 1–46, 2015.

[15] C. Mastroianni, M. Meo, and G. Papuzzo, 'Reinforcement learning for energy-aware resource allocation in cloud computing systems,' IEEE Trans. Netw. Serv. Manag., vol. 20, no. 3, pp. 2845–2859, 2023.

[16] H. Sun, P. Stolf, J. M. Pierson, and G. da Costa, 'Neural network-assisted metaheuristics for green cloud resource scheduling,' Future Gener. Comput. Syst., vol. 150, pp. 234–249, 2024.

[17] X. Li, P. Garraghan, X. Jiang, Z. Wu, and J. Xu, 'Holistic virtual machine scheduling in cloud datacenters towards minimizing total energy,' IEEE Trans. Parallel Distrib. Syst., vol. 29, no. 6, pp. 1317–1331, 2018.

[18] W. Chen, X. Qiao, J. Wei, and T. Huang, 'Graph neural networks for VM placement,' in Proc. ACM SoCC, 2023, pp. 312–325.

[19] N. Singh and H. Hachimi, 'Hybrid metaheuristics in cloud resource management: A comprehensive survey,' Concurr. Comput. Pract. Exp., vol. 34, no. 15, e6920, 2022.

[20] S. Ilager, K. Ramamohanarao, and R. Buyya, 'ETAS: Energy and thermal-aware dynamic VMC with proactive hotspot mitigation,' Concurr. Comput. Pract. Exp., vol. 35, no. 3, e7542, 2023.

[21] M. H. Nadimi-Shahraki, S. Taghian, and S. Mirjalili, 'An improved grey wolf optimizer for solving engineering problems,' Expert Syst. Appl., vol. 166, Art. no. 113917, 2021.

[22] E. Emary, W. Yamany, A. E. Hassanien, and V. Snasel, 'Multi-objective gray-wolf optimization for attribute reduction,' Procedia Comput. Sci., vol. 65, pp. 623–632, 2015.

[23] S. Zhang, Y. Zhou, Z. Li, and W. Pan, 'Grey wolf optimizer for unmanned combat aerial vehicle path planning,' Adv. Eng. Softw., vol. 99, pp. 121–136, 2016.

[24] C. Lu, L. Gao, X. Li, and S. Xiao, 'A hybrid multi-objective grey wolf optimizer for dynamic scheduling in a real-world welding industry,' Eng. Appl. Artif. Intell., vol. 57, pp. 61–79, 2017.

[25] S. Gupta and K. Deep, 'A novel random walk grey wolf optimizer,' Swarm Evol. Comput., vol. 44, pp. 101–112, 2019.

[26] A. A. Heidari et al., 'Harris hawks optimization: Algorithm and applications,' Future Gener. Comput. Syst., vol. 97, pp. 849–872, 2019.

[27] S. Malik, D. Kim, and P. Singh, 'An improved chaotic grey wolf optimizer for engineering design problems,' Soft Comput., vol. 25, no. 14, pp. 9339–9360, 2021.

[28] E. Emary, H. M. Zawbaa, and A. E. Hassanien, 'Binary grey wolf optimization approaches for feature selection,' Neurocomputing, vol. 172, pp. 371–381, 2016.

[29] N. Singh and S. B. Singh, 'Hybrid algorithm of particle swarm optimization and grey wolf optimizer for improving convergence performance,' J. Appl. Math., vol. 2017, Art. no. 2030489.

[30] N. Mittal, U. Singh, and B. S. Sohi, 'Modified grey wolf optimizer for global engineering optimization,' Appl. Comput. Intell. Soft Comput., vol. 2016, Art. no. 7950348.

[31] R. Kaveh and M. Dadkhah, 'Opposition-based grey wolf optimizer with population initialization using Quasi-opposition,' Knowl.-Based Syst., vol. 196, Art. no. 105764, 2020.

[32] A. A. Heidari and P. Pahlavani, 'An efficient modified grey wolf optimizer with Lévy flight for optimization tasks,' Appl. Soft Comput., vol. 60, pp. 115–134, 2017.

[33] S. Mirjalili, S. Saremi, S. M. Mirjalili, and L. D. S. Coelho, 'Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization,' Expert Syst. Appl., vol. 47, pp. 106–119, 2016.

[34] S. A. El-Said, 'An optimized grey wolf optimizer for workflow scheduling in cloud environment,' Cluster Comput., vol. 24, pp. 2273–2292, 2021.

[35] L. Abualigah and A. Diabat, 'Advances in sine cosine algorithm: A comprehensive survey,' Artif. Intell. Rev., vol. 54, pp. 2567–2608, 2021.

[36] D. Meisner, B. T. Gold, and T. F. Wenisch, 'PowerNap: Eliminating server idle power,' in Proc. ASPLOS XIV, 2009, pp. 205–216.

[37] X. Fan, W. D. Weber, and L. A. Barroso, 'Power provisioning for a warehouse-sized computer,' in Proc. ISCA, 2007, pp. 13–23.

[38] A. Beloglazov, 'Energy-Efficient Management of Virtual Machines in Data Centers for Cloud Computing,' Ph.D. dissertation, Univ. of Melbourne, 2013.

[39] J. Chen, X. Wang, and M. Steurer, 'Carbon-aware cloud scheduling: A comprehensive survey,' IEEE Commun. Surv. Tutor., vol. 27, no. 1, pp. 125–167, 2025.

[40] J. Moore, J. Chase, P. Ranganathan, and R. Sharma, 'Making scheduling cool: Temperature-aware workload placement in data centers,' in Proc. USENIX ATC, 2005, pp. 61–75.

[41] X. Meng, V. Pappas, and L. Zhang, 'Improving the scalability of data center networks with traffic-aware virtual machine placement,' in Proc. IEEE INFOCOM, 2010, pp. 1–9.

[42] A. Ghodsi et al., 'Dominant resource fairness: Fair allocation of multiple resource types,' in Proc. USENIX NSDI, 2011, pp. 323–336.

[43] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and R. Buyya, 'CloudSim: A toolkit for modeling and simulation of cloud computing environments,' Softw. Pract. Exp., vol. 41, no. 1, pp. 23–50, 2011.

[44] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, 'A fast and elitist multiobjective genetic algorithm: NSGA-II,' IEEE Trans. Evol. Comput., vol. 6, no. 2, pp. 182–197, Apr. 2002.