

Comparative Analysis of Classical, Metaheuristic, and Reinforcement Learning Approaches to Load Balancing in Cloud Computing

Mahesh Loya^{1*}, Punam Shinde²

Abstract

Efficient load balancing remains one of the most critical challenges in cloud computing, particularly as modern cloud environments rely on heterogeneous virtual machines (VMs) with varying computational capabilities. The effective distribution of workloads across these resources directly influences system performance, energy efficiency, and service reliability. This paper presents a comparative evaluation of three major categories of load-balancing approaches: traditional scheduling heuristics, nature-inspired metaheuristic algorithms, and reinforcement learning (RL)-based techniques. The analysis is conducted across five key performance dimensions, including response latency, throughput, energy consumption, resource utilization, and fault resilience. A systematic review of ten peer-reviewed studies published between 2019 and 2024 was performed to identify current trends, strengths, and limitations of existing approaches. The findings indicate that RL-based schedulers have achieved significant improvements in simulated cloud environments, delivering energy savings of up to 56% and reducing response times by as much as 46% when compared with optimized conventional methods. However, these benefits are accompanied by substantial computational overhead during training, often requiring GPU-accelerated infrastructure. Additionally, RL models face scalability challenges due to state-space explosion when the number of virtual machines increases beyond several hundred nodes. The review further reveals that hybrid approaches, which combine the exploration capabilities of metaheuristic algorithms with the adaptive decision-making strengths of reinforcement learning, consistently produce superior performance across multiple evaluation metrics. Despite recent progress, several important research gaps remain unresolved. These include the lack of standardized benchmarking frameworks, limited integration of energy-aware scheduling objectives, inadequate fault-recovery mechanisms, and poor scalability of state representations. To address these challenges, this paper proposes a future research agenda focused on lightweight, scalable, and transfer-learning-enabled reinforcement learning agents capable of supporting next-generation cloud computing environments more efficiently and reliably.

*Author for Correspondence

Mahesh Loya

E-mail: loyamahesh3@gmail.com

¹Student, Department of Information Technology, Pune Institute of Computer Technology, Pune, Maharashtra, India

²Professor, Department of Information Technology, Pune Institute of Computer Technology, Pune, Maharashtra, India

Received Date: April 03, 2026

Accepted Date: July 03, 2026

Published Date: August 10, 2026

Citation: Mahesh Loya, Punam Shinde. Comparative Analysis of Classical, Metaheuristic, and Reinforcement Learning Approaches to Load Balancing in Cloud Computing. Journal of Operating Systems Development & Trends. 2026; 13(2): 1–6p

Keywords: Cloud computing, load balancing, reinforcement learning, metaheuristic optimization, energy-aware scheduling, virtual machine allocation

INTRODUCTION

Modern cloud data centers host thousands of virtual machines across geographically distributed regions, serving workloads ranging from video transcoding to real-time financial processing. Poor task allocation causes some VMs to stall under excess demand, whereas others sit idle, wasting energy and violating service-level agreements

(SLAs). Therefore, load balancing must optimize several conflicting objectives, such as latency, throughput, energy consumption, and cost, all under uncertainty regarding future arrivals.

Three algorithm families address this issue. Classical heuristics (Round Robin, Least Connections, Min-Min, Max-Min) assign tasks through fixed rules at a negligible cost. Nature-inspired metaheuristics (PSO, ABC, ACO) explore larger solution spaces via iterative search but may themselves become a bottleneck. RL agents learn allocation policies by interacting with the system over time, adapting to workload shifts, but incurring training overhead. This review synthesizes ten studies published between 2019 and 2024 and addresses three questions: (i) when do classical schedulers match RL alternatives? (ii) What gains and costs do hybrid metaheuristic and RL frameworks introduce? (iii) Which barriers prevent RL-based balancing from reaching production?

SCOPE, METHODOLOGY AND RESEARCH QUESTIONS

Task-to-VM assignment is NP-hard when real-time constraints are present. An ideal scheduler would minimize the makespan, maintain safe utilization, limit power draw, and recover from failures simultaneously. Classical policies guarantee fairness with near-zero latency but cannot react to traffic spikes or virtual machine (VM) degradation. Metaheuristics add scheduling delays, and RL agents require GPU-intensive training. The central question is: At what workload variability level does the overhead of intelligent scheduling pay for itself? This review is restricted to IaaS-layer task-to-VM scheduling; network-layer balancing and container orchestration fall outside its scope.

Ten papers were selected from IEEE Xplore, Springer, and ScienceDirect using keywords including "cloud load balancing," "reinforcement learning scheduling," and "metaheuristic optimization." The analysis addresses: RQ1: At what workload dynamism do classical heuristics underperform RL alternatives? RQ2: Do hybrid metaheuristic plus RL frameworks justify their complexity? RQ3: What principal barriers prevent RL schedulers from production deployment?

Analytical Framework

All ten studies were evaluated against five dimensions: Response Latency (submission-to-completion time), throughput (tasks completed per unit time), Energy Consumption (total power including idle VMs), resource utilization (fraction of CPU/memory actively serving tasks), and Fault Resilience (performance maintenance under VM failures or network partitions). Figure 1 presents the taxonomy of the reviewed strategies.

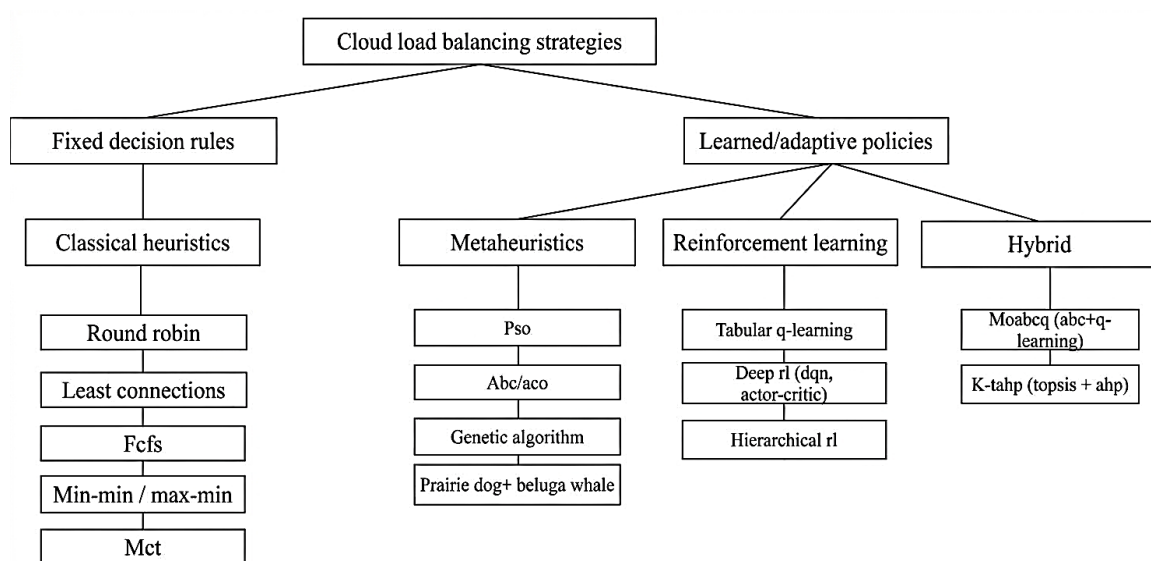


Figure 1. Hierarchical classification of cloud load balancing strategies organized by decision mechanism.

Table 1. Performance characteristics of classical scheduling heuristics.

Heuristic	When it works well	When it fails
Round Robin	Uniform VMs, steady arrival rate	Heterogeneous VMs, bursty traffic
Min-Min	Predominantly short tasks	Long tasks indefinitely delayed
Max-Min	Mixed workloads, few large tasks	Highly variable task sizes
FCFS	Low contention, fair ordering	Convoy effect from large tasks
MCT	Known task to VM profiles	Profiles unavailable or stale

CLASSICAL SCHEDULING HEURISTICS (RQ1)

Round Robin, Least Connections, FCFS, Min-Min, Max-Min, and MCT depend only on the current queue or VM state with no memory of the past outcomes. Gamal et al. [1] observed that Round Robin achieved near-perfect fairness across homogeneous pools, but performance dropped sharply when VM capacities differed by more than 2x. Souravlas et al. [2] found that their Markov-process model outperformed FCFS by 18 to 23% by accounting for heterogeneous processing rates. Classical heuristics perform within 5–10% of optimal when tasks are uniform, VMs are uniform, and arrival rates are stationary. The violation of any condition causes nonlinear degradation. Table 1 summarizes these trade-offs.

METAHEURISTIC AND BIO-INSPIRED OPTIMIZATION

Nature-inspired algorithms find solutions within 5–15% of the theoretical optima in polynomial time. Gamal et al. [1] combined ABC and ACO into an "osmotic" framework that cut unnecessary VM migrations by 30 to 40%, with gradual rather than binary migration decisions. Ramya and Ayothi [3] fused Prairie Dog and Beluga Whale Optimization, achieving consistent Pareto improvements in CloudSim but requiring seven hyperparameters. Gao et al. [4] embedded AHP+TOPSIS multi-criteria analysis directly into the Kubernetes pod scheduler (K-TAHP), achieving up to 60% efficiency gains. This suggests that the bottleneck may lie in problem framing rather than algorithm sophistication. A common limitation is that nearly all experiments use CloudSim or comparable simulators that lack network congestion, storage I/O contention, and multi-tenant interference modelling; simulation gains may shrink under realistic conditions.

REINFORCEMENT LEARNING AGENTS FOR DYNAMIC SCHEDULING

RL schedulers cast load balancing as a Markov Decision Process (MDP): the agent observes the system state, selects a task-to-VM assignment, and receives a reward tied to latency, energy, or SLA compliance, converging on a near-optimal policy without an explicit environment model. Figure 2 illustrates this feedback loop.

Lahande et al. [5] applied tabular Q-learning to a moderate-scale simulation, improving the utilization and energy efficiency over static baselines by exploiting temporal workload patterns. Jayanetti et al. [6] extended this to deep RL with a hierarchical state-space formulation, achieving 56% energy reduction and 46% execution time reduction, although training required GPU hardware and thousands of episodes. Mohammadian et al. [7] found that most RL formulations treat VM failures as external disturbances rather than encoding failure probability in the state, leading to poor reassignment during cascades. Kruekaew and Kimpan [8] embedded Q-learning within an ABC loop (MOABCQ), reducing convergence time by 25 to 35% compared to standalone ABC while sidestepping the DRL state-space explosion. Gures et al. [9] surveyed ML techniques for heterogeneous networks, positioning RL as a family that generates its own training signal through environmental interaction.

The key distinction between the RL families lies in their scalability and training cost. Tabular Q-learning is effective for moderate-scale deployments but suffers from the curse of dimensionality. Deep RL scales better through function approximation but require a GPU infrastructure. Hybrid approaches that embed RL within metaheuristic loops offer a practical compromise between these two extremes.

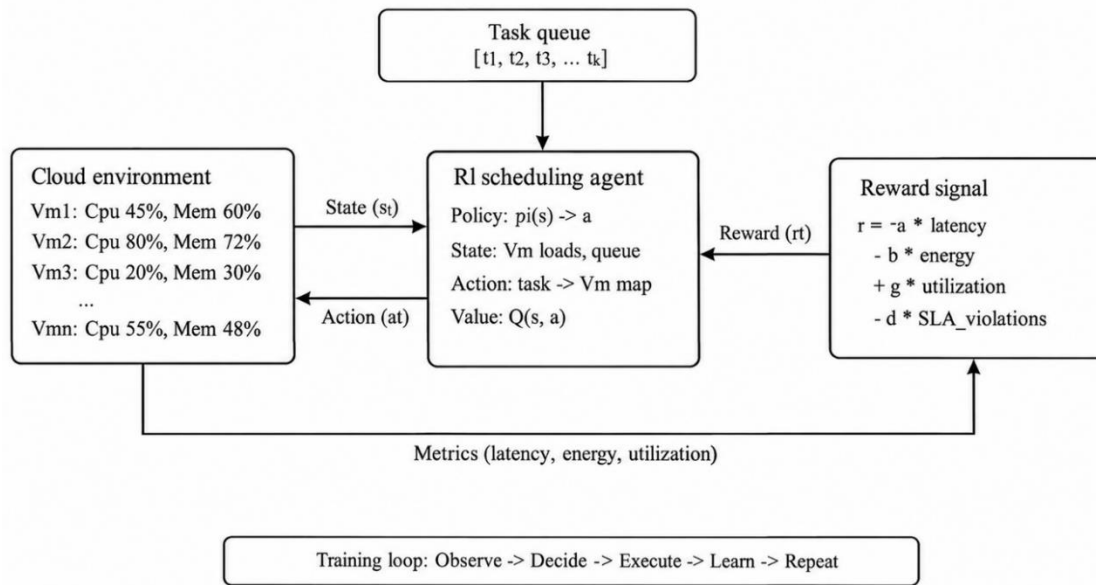


Figure 2. Feedback loop of an RL-based load balancing agent showing the observe, decide, execute, and learn cycle.

CROSS-METHOD COMPARISON (RQ2)

The hybrid metaheuristic plus RL methods produced the strongest multi-objective results, as shown in Table 2. In MOABCQ [8], the metaheuristic provides a broad exploration that prevents premature convergence, whereas the RL component exploits the learned structure to speed up convergence. However, hybrid systems require the maintenance of two optimization loops with synchronized state representations. The framework proposed by Ramya and Ayothi [3] involves seven hyperparameters; however, none of the reviewed studies implemented automated tuning. In contrast, Gao et al. [4] showed that K-TAHP, a deterministic multi-criteria analysis method, achieved competitive efficiency gains through a parameter-free process.

BARRIERS TO PRODUCTION DEPLOYMENT (RQ3)

State-space scalability: The RL observation space dimensionality increases with the product of the task types and VM count. Lahande et al. [5] observed degradation beyond approximately 50 VMs with tabular Q-learning; Jayanetti et al. [10] tested 250 servers across 10 data centers, which is still far below hyperscale deployments.

Benchmark standardization: Eight of ten studies relied on CloudSim, which omits multi-tenant interference, topology effects, and storage contention. Without standardized real-infrastructure benchmarks, cross-study comparisons remain unreliable.

Table 2. Comparative assessment of load balancing strategies across five performance dimensions.

Strategy	Latency	Throughput	Energy	Res. Util.	Fault R.	Refs
Classical (RR, LC, FCFS)	Low cost; poor heterog.	Adequate steady	Not optimized	Moderate	None	[1, 2]
Metaheur. (ABC, ACO, PSO)	Adds delay	Improved	Notable red.	High	Limited	[1, 4, 3]
Tabular RL	Low infer.; mod. train	Adapts	Better than static	High	Fragile	[8, 5]
Deep RL	Lowest trained	Best dynamic	Up to 56% red.	Highest	Under-expl.	[9, 6]
Hybrid Meta.+RL	Moderate	Best overall	Strong balance	Near optimal	Promising	[8, 4, 3]

Energy-awareness integration: Only three studies [1, 3, 6] included energy as a first-class optimization objective. Given that data-center electricity is approaching 4% of global production, treating energy as a soft constraint is no longer sufficient to achieve sustainability.

Fault recovery: Mohammadian et al. [8] found that most RL formulations lack failure-handling mechanisms. Incorporating failure prediction into state representation remains an open problem.

These barriers are closely related to one another: scalability improvements require realistic benchmarks for validation, energy-aware policies need fault-tolerant foundations, and transfer learning depends on standardized evaluation across different environments.

FUTURE DIRECTIONS AND RESEARCH GAPS

Lightweight RL with transfer learning: Pre-training in simulated environments followed by production fine-tuning can substantially reduce training costs. Real-infrastructure benchmarking: A shared platform on OpenStack or CloudLab with reproducible traces would help bridge the gap between simulation and production. Joint energy and SLA optimization: Multi-objective RL using Pareto-front reward shaping should co-optimize energy and SLA compliance. Self-healing agents: Integrating scheduling with failure-prediction models would enable proactive redistribution before degradation occurs.

Key gaps include RL scalability limitations from high-dimensional state spaces, over-reliance on simulation without real-world validation, inconsistent energy-aware integration, and underdeveloped fault tolerance.

CONCLUSION

This review evaluated three generations of cloud load balancing strategies against five performance dimensions using evidence from ten studies published between 2019 and 2024. Classical heuristics suit homogeneous, steady-state environments in which near-zero overhead outweighs adaptability. Metaheuristics extend the optimization space at the cost of scheduling delay and hyperparameter tuning. RL agents offer the strongest adaptability; deep RL achieved up to 56% energy reduction and 46% latency reduction, although training and scalability constraints confine them to moderate deployments. The hybrid metaheuristic plus RL frameworks consistently outperformed either family in isolation, combining broad exploration with targeted exploitation. The four identified barriers, namely state-space scalability, benchmark standardization, energy-awareness integration, and fault recovery, mark the critical path from simulation to production deployment.

REFERENCES

1. Kruekaew B, Kimpan W. Multi-objective task scheduling optimization for load balancing in cloud computing environment using hybrid artificial bee colony algorithm with reinforcement learning. IEEE Access. 2022;10:17803-18.
2. Gures E, Shayea I, Ergen M, Azmi MH, El-Saleh AA. Machine learning-based load balancing algorithms in future heterogeneous networks: a survey. IEEE Access. 2022;10:37689-717.
3. Gamal M, Rizk R, Mahdi H, Elnaghi BE. Osmotic bio-inspired load balancing algorithm in cloud computing. IEEE Access. 2019;7:42735-44.
4. Souravlas S, Anastasiadou SD, Tantalaki N, Katsavounis S. A fair, dynamic load balanced task distribution strategy for heterogeneous cloud platforms based on Markov process modeling. IEEE Access. 2022;10:26149-62.
5. Gao R, Xie X, Guo Q. K-TAHP: a Kubernetes load balancing strategy based on TOPSIS+AHP. IEEE Access. 2023;11:102132-9.
6. Ramya K, Ayothi S. Hybrid prairie dog and beluga whale optimization algorithm for multi-objective load balanced-task scheduling in cloud computing environments. China Commun. 2024;21(7):307-24.

-
7. Lahande PV, Kaveri PR, Saini JR, Kotecha K, Alfarhood S. Reinforcement learning approach for optimizing cloud resource utilization with load balancing. *IEEE Access*. 2023;11:127567-77.
 8. Mohammadian V, Navimipour NJ, Hosseinzadeh M, Darwesh A. Fault-tolerant load balancing in cloud computing: a systematic literature review. *IEEE Access*. 2022;10:12714-31.
 9. Jayanetti A, Halgamuge S, Buyya R. Deep reinforcement learning for energy and time optimized scheduling of precedence-constrained tasks in edge-cloud computing environments. *Future Gener Comput Syst*. 2023;137:14-30.
 10. Jayanetti A, Halgamuge S, Buyya R. Reinforcement learning based workflow scheduling in cloud and edge computing environments: a taxonomy, review and future directions [Preprint]. 2024. arXiv:2408.02938.