Cloud-Based Replication Models Using AI Techniques for Enhanced Data Management

Moneef M. Jazzar, Aws I. Abueid*
Faculty of Computing Studies, Arab Open University, Kuwait

***Abstract*—Elastic cloud infrastructure relies on dynamic replication mechanisms to maintain service availability and performance under fluctuating, non-stationary workloads. However, conventional threshold-based and static replication strategies frequently fail to maintain latency stability and Service Level Agreement (SLA) compliance in highly dynamic environments characterised by bursty, peak-stress traffic. This study introduces a Q-learning–based adaptive replication framework that formulates replication control as a sequential decision-making problem. The system is modelled as a Markov Decision Process (MDP), where replication adjustments are selected to maximise cumulative discounted reward, integrating latency minimisation, SLA violation penalties, and replica cost regularisation within a unified optimisation objective. A controlled cloud simulation environment was developed to emulate phased stochastic workload patterns, including normal, burst, sustained peak, and recovery intervals. The reinforcement learning controller was trained over 5000 episodes and subsequently evaluated under fixed-policy conditions against a reaction-delayed rule-based baseline controller. Experimental results demonstrate substantial improvements in performance stability. The proposed learning-based controller achieves a significant reduction in average latency, strong suppression of 95th percentile tail latency, and complete elimination of SLA violations under dynamic workload conditions. Unlike reactive threshold-based mechanisms, the learned policy anticipates workload transitions and proactively adjusts replication levels through long-term reward optimisation. These findings confirm that learning-driven replication control provides a structurally superior paradigm for latency-sensitive elastic cloud systems. By embedding SLA awareness directly into the reward formulation, replication management is transformed from a static configuration task into an adaptive, intelligent control process.**

Keywords*—*Q-learning; elastic cloud replication; SLA-aware control; latency optimization; adaptive replication; reinforcement learning

I. Introduction

Elastic cloud computing has become the foundational infrastructure for modern distributed data management systems, enabling scalable replication across geographically dispersed data centres. While elasticity enhances availability and resource flexibility, it also introduces substantial performance variability. In latency-sensitive environments—such as interactive cloud services, distributed ledgers, and real-time data platforms—maintaining stable response time under fluctuating workloads and heterogeneous network conditions remains a critical challenge.

At the replication-protocol level, recent advances in low-latency Byzantine Fault Tolerant (BFT) mechanisms demonstrate that reducing the number of communication rounds and enabling optimistic execution paths can significantly decrease end-to-end delay while preserving consistency guarantees [1]. These findings highlight the importance of communication complexity and fast-path execution in wide-area deployments. However, protocol-level optimisation alone does not eliminate latency bottlenecks, particularly when workload intensity varies dynamically.

*Corresponding author.

Storage-layer restructuring has been shown to substantially reduce execution delay even when replication protocols remain unchanged, reinforcing the impact of state-access architecture on end-to-end performance [2]. Similarly, communication-centric redesign in distributed infrastructures reveals that synchronisation imbalance and data-movement overhead frequently dominate latency behaviour, particularly under cloud-like network conditions [3]. These studies collectively confirm that replication performance depends on coordinated optimisation across multiple system layers rather than isolated protocol enhancements.

In elastic cloud environments, replication and scaling decisions inherently involve multi-objective tradeoffs. Minimising latency must be balanced with resource utilisation, operational costs, and system resilience. Multi-objective optimisation frameworks demonstrate that response time and resource efficiency are competing objectives requiring Pareto-aware decision strategies rather than single-metric tuning [4]. From a Quality of Service (QoS) perspective, dynamic service placement and provisioning mechanisms emphasise that latency constraints must be enforced explicitly as Service Level Agreement (SLA) requirements under volatile workload conditions [5].

Recent developments in AI-driven orchestration suggest that intelligent control agents can provide adaptive decision-making capabilities beyond traditional rule-based management [6]. Learning-based approaches are particularly effective for modelling nonlinear and time-varying system behaviour across distributed environments [8], while structured AI deployment frameworks underscore the importance of measurable performance improvements in real-world infrastructures [7]. Despite these advances, existing work typically addresses protocol efficiency, storage optimisation, resource orchestration, or SLA monitoring independently. Few studies integrate SLA constraints, adaptive replication control, and learning-based decision-making within a unified latency-centric framework.

Motivated by this gap, this study formulates elastic cloud replication as a sequential decision-making problem. Instead of treating latency as a passive monitoring metric, it is modelled as an explicit control variable within a reinforcement learning objective. Replication degree adjustment is formulated as a Markov Decision Process (MDP), where decisions are optimised to minimise latency, penalise SLA violations, and regulate replica cost simultaneously by maximising cumulative discounted rewards.

To operationalize this formulation, a tabular Q-learning–based adaptive replication controller is implemented and evaluated in a controlled geo-distributed cloud simulation environment. The learning agent dynamically adjusts replication degree in response to phased stochastic workload patterns, including normal, burst, peak, and recovery intervals. Performance is compared against a reaction-delayed rule-based baseline controller to isolate the impact of learning-driven adaptation.

The contributions of this study are as follows:

- A unified SLA-aware elastic replication framework formulated as a discrete-time MDP.
- A mathematically grounded reinforcement learning objective integrating latency minimisation, violation penalties, and replica cost regularisation.
- An implemented Q-learning adaptive controller was evaluated under reproducible workload conditions.
- A controlled comparative analysis demonstrating substantial improvements over reaction-based replication strategies.

Experimental results confirm that the proposed learning-based controller significantly reduces average and tail latencies, while eliminating SLA violations under bursty workload conditions. These findings demonstrate that embedding SLA-awareness directly into the reinforcement learning objective transforms replication management from a reactive threshold mechanism into an adaptive intelligent control process for latency-sensitive elastic cloud systems. Experimental results confirm that the proposed learning-based controller significantly reduces average and tail latencies while eliminating SLA violations under bursty workload conditions.

Unlike existing replication approaches that treat latency as a monitored metric, this work formulates replication as a sequential decision-making problem and embeds SLA constraints directly into the learning objective.

II. Literature Review

A. AI in Distributed Replication Control

Artificial intelligence has increasingly been incorporated into distributed data management architectures to enhance structural decision-making and system adaptability. AI-assisted frameworks for dynamic fragmentation, allocation, and replication have demonstrated improved efficiency and reliability in distributed database systems, even when workload characteristics are partially unknown during the design stage [9]. These approaches highlight the potential of intelligent control mechanisms to optimise structural decisions and reduce manual configuration overhead.

However, most existing AI-assisted replication models primarily target database-level optimisation rather than latency-sensitive elastic cloud environments. They often focus on data placement and fragmentation efficiency without explicitly modelling Service Level Agreement (SLA) constraints or latency dynamics under fluctuating workloads. Moreover, reinforcement learning–based closed-loop adaptive replication control remains largely unexplored in these contexts.

B. Latency-Aware Replica Placement

Latency-bound replica management strategies have been proposed for decentralised edge–cloud systems, where replica placement decisions aim to minimise delay while reducing coordination and resource overhead [10]. These models emphasise proximity-based placement, communication efficiency, and decentralised coordination to stabilise response time in distributed infrastructures.

Although latency is explicitly considered in these approaches, it is typically treated as a constrained performance metric rather than as an optimisation variable within a sequential adaptive control framework. Furthermore, many of these strategies rely on deterministic rules or heuristic placement algorithms rather than learning-based controllers that can adapt to non-stationary workload phases and evolving system states.

C. Multi-Objective Optimisation in Elastic Clouds

Elastic cloud replication inherently involves competing objectives, including minimising latency, reducing storage overhead, improving resource utilisation efficiency, and ensuring availability. Multi-objective optimisation frameworks confirm that response time must be jointly optimised with operational cost and resource balancing rather than treated independently [11]. Pareto-based approaches provide structured mechanisms to manage these trade-offs under dynamic cloud conditions.

Despite these advancements, many multi-objective models operate primarily at the scheduling or provisioning layer and do not explicitly account for adjusting the replication degree in SLA-constrained elastic environments. In addition, optimisation strategies are frequently static, heuristic-based, or rule-driven rather than policy-learning-based. The integration of multi-objective trade-offs into a reinforcement learning–driven replication policy remains insufficiently investigated.

D. Elastic Auto-Scaling and Adaptive Resource Control

Auto-scaling mechanisms dynamically adjust computational capacity to maintain performance stability in the face of workload variability [12]. Learning-based scaling strategies have demonstrated greater responsiveness than static threshold-based policies, particularly in bursty-demand scenarios. However, conventional auto-scaling approaches predominantly focus on computational resource allocation (e.g., virtual machine scaling or container orchestration) rather than replication control. The interaction between replication degree, workload dynamics, and SLA compliance is rarely modelled as a unified sequential decision-making problem. Consequently, replication and scaling policies are often optimised independently rather than coordinated within a single adaptive control framework.

E. SLA-Aware Monitoring and Proactive Management

SLA-aware systems incorporate monitoring and anomaly detection mechanisms to identify performance degradation before contractual thresholds are violated [13]. Machine learning–based monitoring frameworks enable proactive detection of anomaly patterns and potential SLA risks. Nevertheless, most SLA-aware solutions operate at the diagnostic or supervisory layer. SLA compliance is typically treated as an output metric or a post-hoc evaluation criterion rather than embedded directly into the optimisation objective. Few systems integrate SLA-violation penalties into the control policy itself, limiting their ability to perform proactive adaptation via learning-based optimisation.

Recent advances in multi-objective optimisation and intelligent system design further reinforce the importance of adaptive control in distributed environments. Differentiable architecture search techniques have demonstrated the effectiveness of jointly optimising competing objectives such as performance and latency, highlighting the importance of automated decision optimisation under resource constraints [14]. Similarly, Pareto-based optimisation approaches have been successfully applied to fog–cloud task offloading, enabling systematic trade-offs between latency, resource utilisation, and operational cost in distributed infrastructures [15].

In addition, reinforcement learning has been effectively applied to dynamic replication and reliability optimisation, where the number of replicas is adaptively adjusted based on runtime system conditions, demonstrating the effectiveness of learning-based control policies in improving system robustness and efficiency [16].

From an SLA perspective, foundational work has established formal frameworks for SLA evaluation and composition in service-oriented architectures [17], while recent studies emphasise the need for dynamic and adaptive SLA management in cloud and fog environments characterised by high variability and non-stationary workloads [18]. These developments collectively indicate a shift toward intelligent, optimisation-driven, and SLA-aware adaptive system management.

F. Research Gap and Motivation

Although substantial progress has been achieved in cloud replication, latency optimisation, auto-scaling, and SLA monitoring, these research directions are predominantly pursued in isolation. Existing solutions tend to optimise individual system components—such as replica placement, cost efficiency, or anomaly detection—without integrating them into a unified adaptive control architecture capable of holistic decision-making. In elastic cloud environments, replication management is fundamentally a sequential control problem. The replication degree directly influences latency, infrastructure cost, and SLA compliance over time. Nevertheless, most current approaches treat latency as a passive performance indicator rather than as an actively controllable decision variable. Replication adjustments are typically triggered by predefined thresholds or heuristic rules, leading to reactive behaviour, delayed adaptation, and instability under bursty or non-stationary workload conditions.

Furthermore, multi-objective trade-offs between latency minimisation and cost efficiency are commonly addressed through static optimisation or rule-based heuristics, rather than through adaptive policy learning. SLA-awareness is frequently implemented at the monitoring layer, functioning as a diagnostic mechanism rather than being embedded directly into the optimisation objective. Most critically, reinforcement learning–based closed-loop control for dynamic replication adjustment remains insufficiently explored in latency-sensitive elastic cloud systems. Few existing studies formulate replication as a Markov Decision Process in which control actions explicitly influence future system states and long-term cumulative performance.

This research gap motivates the proposed framework, which formulates replication control as a sequential optimisation problem and addresses it via Q-learning. By embedding latency, cost regularisation, and SLA violation penalties directly into the reward function, the model enables proactive, learning-driven replication management that adapts to non-stationary workload dynamics and minimises long-term performance degradation.

III. Problem Formulation and Q-Learning–based Adaptive Replication Model

A. System Model and Objective

In elastic cloud environments, the number of active replicas directly influences response latency, SLA compliance, and infrastructure cost. Let:

- R(t) denotes the number of active replicas at time step t
- W(t) denotes workload intensity
- L(t) denotes observed latency
- L_max denotes the SLA latency threshold

Latency is modelled as inversely proportional to replication capacity and proportional to workload intensity:

Latency is modelled as inversely proportional to replication capacity and proportional to workload intensity [see Eq. (1)]:

$$L(t) = \frac{W(t)}{R(t)} + \varepsilon_t \tag{1}$$

where, $\varepsilon_t \sim \mathcal{N}(0, \sigma^2)$ represents stochastic network noise. An SLA violation indicator is defined as Eq. (2):

$$\mathrm{V(t)} = \begin{cases} 1, & \text{if } \mathrm{L(t)} > \mathrm{L_{max}} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

The control objective is to adjust dynamically. R(t) in order to:

- Minimise average latency
- Reduce SLA violations
- Avoid excessive resource provisioning

This yields a sequential decision-making problem under uncertainty.

B. Markov Decision Process Formulation

The adaptive replication problem is modelled as a discrete-time Markov Decision Process (MDP), defined by the tuple.

$$(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma)$$

where,

- $\mathcal{S}$= state space
- $\mathcal{A}$= action space
- $\mathcal{P}$= state transition dynamics
- r= reward function
- $\gamma \in (0,1)$= discount factor

1) State space: The system state at time t is defined as Eq. (3):

$$\mathrm{S(t) = (R(t), W_{bin}(t), V(t))} \tag{3}$$

To enable tabular Q-learning, the workload is discretised [see Eq. (4)]:

$$\mathrm{W_{bin}(t)} = \left\lfloor \frac{\mathrm{W(t)}}{\Delta_{\mathrm{W}}} \right\rfloor \tag{4}$$

where,

- Δ_{W}= workload discretisation interval
- $\lfloor \cdot \rfloor$= floor operator

In the implemented simulation: $\Delta_{\mathrm{W}} = 30$
This discretisation reduces state dimensionality while preserving phase information about the workload.

2) Action space: At each time step, the agent selects [see Eq. (5)]:

$$\mathrm{A(t)} \in \{-1, 0, +1\} \tag{5}$$

where,

- $-1 \rightarrow$ decrease replica count
- $0 \rightarrow$ maintain current replicas
- $+1 \rightarrow$ increase replica count

Replica update rule [see Eq. (6)]:

$$\mathrm{R(t+1) = min\,(R_{max}, max\,(1, R(t) + A(t)))} \tag{6}$$

where, $\mathrm{R_{max}}$ is the maximum allowed replication degree.

C. Reward Function Design

The reward function integrates three control objectives:

- Latency minimisation
- SLA violation penalty
- Replica cost regularisation

The immediate reward at the time t is defined as Eq. (7):

$$r(t) = -\frac{L(t)}{L_{\text{norm}}} - \lambda_v V(t) - \lambda_r R(t) \tag{7}$$

where,

- $L_{\text{norm}}$is a latency normalisation constant
- $\lambda_v$is the SLA violation penalty coefficient
- $\lambda_r$is the replica cost penalty coefficient

This reward formulation ensures that:

- Higher latency reduces the reward value
- SLA violations incur strong penalties
- Excessive replication is discouraged

By embedding latency and violation penalties directly into the reward structure, the model transforms SLA compliance from a monitoring constraint into an optimisation objective. Consequently, replication control becomes a reward-driven adaptive process within the reinforcement learning loop.

D. Q-Learning Algorithm

The Q-learning algorithm estimates the optimal action-value function $Q(S, A)$.

The update rule is Eq. (8):

$$\mathrm{Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[r_t + \gamma \max_{a \in \mathcal{A}} \; Q(S_{t+1}, a) - Q(S_t, A_t) \right]} \tag{8}$$

where,

- α= learning rate
- γ= discount factor

The learned policy is defined as Eq. (9):

$$\pi(S) = \arg \max_{a \in \mathcal{A}} Q(S, a) \tag{9}$$

This policy selects the action with the highest expected long-term cumulative reward.

E. Training Configuration

The agent was trained using:

- 5000 episodes
- 400 time steps per episode
- Learning rate $\alpha = 0.1$
- Discount factor $\gamma = 0.95$
- ε-greedy exploration strategy

Total episodic reward convergence was monitored to ensure stable learning behaviour.

F. Q-Learning Adaptive Replication Algorithm

To operationalize the mathematical formulation described above, the adaptive replication strategy is implemented using a tabular Q-learning algorithm (see Algorithm 1). The algorithm iteratively updates the action-value function $Q(S, A)$ based on observed rewards and state transitions.

The learning process follows a ε-greedy exploration strategy to balance exploitation of learned policies and exploration of new state–action pairs.

Algorithm 1: Q-Learning Adaptive Replication

```
Initialize
  Input:
    Learning rate $\alpha$
    Discount factor $\gamma$
    Exploration rate $\varepsilon$
    Maximum replicas $R_{max}$

  Initialize Q-table $Q(S, A)$arbitrarily

Compute

For each episode = 1to $N$do
  ├── Initialize environment
  ├── Observe initial state $S$
  └── While (not terminal) do
        ├── Select action $A$using $\varepsilon$-greedy policy
        ├── Execute action $A$
        ├── Observe reward $r$and the next state $S'$
        ├── Update Q-value
        │     $Q(S,A) \leftarrow Q(S,A) + \alpha[r + \gamma \max_a Q(S',a) - Q(S,A)]$
        └── Set $S \leftarrow S'$

End

Return learned policy:

    $\pi(S) = \arg\max_a Q(S,a)$
```

G. Rule-Based Baseline Controller

To isolate the impact of AI-driven learning, a rule-based controller was implemented.

The rule-based strategy:

- Increases replicas when latency exceeds the threshold
- Decreases replicas when latency falls below a safe margin
- Includes a fixed reaction delay
- Unlike Q-learning, the rule-based controller:
- Reacts only after performance degradation
- Does not optimise long-term reward
- Does not anticipate workload transitions

This enables controlled comparative evaluation between adaptive AI control and reactive heuristic control.

IV. Reinforcement Learning–Driven Adaptive Replication Framework

A. Environment Definition

To validate the adaptive replication strategy formulated in Section III, a controlled simulation environment was implemented to emulate a geo-distributed elastic cloud system.

The system evolves in discrete time steps.

$$t = 1, \dots, T.$$

At each time step, the environment is defined by Eq. (10):

$$L(t) = \frac{W(t)}{R(t)} + \varepsilon_t \tag{10}$$

where,

- $W(t)$= workload intensity
- $R(t)$= number of replicas
- $\varepsilon_t \sim \mathcal{N}(0, \sigma^2)$

An SLA violation occurs if [see Eq. (11)]:

$$V(t) = \begin{cases} 1, & L(t) > L_{\max} \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

The replication degree is bounded [see Eq. (12)]:

$$1 \le R(t) \le R_{\max} \tag{12}$$

B. Workload Phases

Workload intensity follows a phased stochastic pattern:

- Normal phase
- Burst phase
- Sustained peak phase
- Recovery phase

Formally [see Eq. (13)]:

$$W(t) = W_{base}(t) + \eta_t \tag{13}$$

where, $\eta_t \sim \mathcal{N}(0, \sigma_W^2)$.

This design creates realistic workload volatility for evaluating adaptive control.

C. Reinforcement Learning Control Policy

The Q-learning agent selects replication adjustment actions [see Eq. (14)]:

$$A(t) \in \{-1, 0, +1\} \tag{14}$$

Replica update [see Eq. (15)]:

$$\mathrm{R(t+1) = \min\,(R_{max}, \max\,(1, R(t) + A(t)))} \tag{15}$$

The reward function is defined as Eq. (16):

$$r(t) = -\frac{L(t)}{L_{norm}} - \lambda_v V(t) - \lambda_r R(t) \tag{16}$$

This ensures:

- Latency reduction

- SLA compliance
- Resource efficiency

D. Learning Objective

The agent maximises expected cumulative discounted reward [see Eq. (17)]:

$$\mathbb{E}\left[\sum_{t=0}^{T}\gamma^{t}\, r(t)\right] \tag{17}$$

where, $\gamma \in (0,1)$.

Policy [see Eq. (18)]:

$$\pi(S) = \arg\max_{a} Q(S,a) \tag{18}$$

Training uses ε-greedy exploration.

E. Rule-Based Baseline

For comparative evaluation, a rule-based controller was implemented.

The baseline:

- Increases replicas when latency exceeds the SLA threshold
- Decreases replicas when latency drops below a safe margin
- Includes reaction delay

Unlike RL, it does not optimise long-term cumulative reward.

V. Experimental Architecture and Workload Model

A. Experimental Architecture

1) Environment overview: To evaluate the theoretical framework introduced in Section 4, the mathematical model is instantiated within a controlled cloud simulation environment. The implementation follows the latency model defined in Eq. (10), the SLA constraint in Eq. (11), and the replication bounds in Eq. (12)

The system evolves in discrete time steps:

where, the simulation horizon is fixed as Eq. (19):

$$T = 400 \tag{19}$$

At each time step, the workload intensity W(t) is generated, the replication degree R(t) is updated by the controller, and the observed latency L(t) is computed.

Latency is modelled as Eq. (20):

$$L(t) = \frac{W(t)}{R(t)} + \varepsilon_t \tag{20}$$

where,

- W(t)denotes workload intensity
- R(t)denotes the number of replicas
- where, $\varepsilon_t \sim \mathcal{N}(0,\sigma^2)$ represents stochastic network noise.

An SLA violation indicator is defined as Eq. (21):

$$V(t) = \begin{cases} 1, & \text{if } L(t) > L_{\max} \\ 0, & \text{otherwise} \end{cases} \tag{21}$$

The replication degree is bounded [see Eq. (22)]:

$$1 \le R(t) \le Rmax \tag{22}$$

This implementation instantiates the theoretical formulation presented in Section IV within a reproducible experimental environment, enabling systematic evaluation of adaptive replication behaviour under controlled stress scenarios.

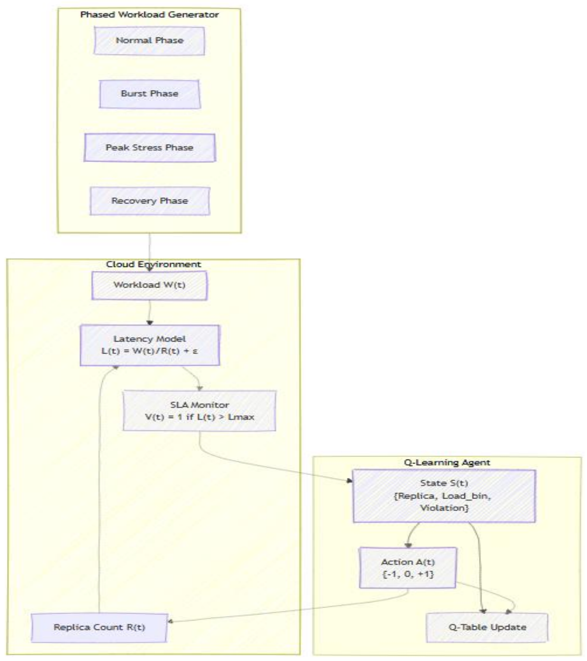

Fig. 1. Q-learning–based adaptive replication framework.

Fig. 1 illustrates the closed-loop architecture. A phased workload generator produces dynamic demand. The cloud environment computes latency as a function of workload and replica count. The SLA monitor detects violations. The Q-learning controller selects replication adjustments. The execution module applies the action and updates the system state, forming a continuous adaptive loop.

2) Architectural components: The architecture consists of:

- Workload Generator – produces phased stochastic demand.
- Cloud Environment – computes latency as a function of workload and replication.
- SLA Monitor – detects violation conditions.
- Q-Learning Agent – learns an optimal replication policy.
- Replica Controller – applies action A(t).

TABLE. I. SYSTEM COMPONENTS DESCRIPTION

Component	***Role***
Workload Generator	Produces phased stochastic demand
Cloud Environment	Computes latency based on replica count
SLA Monitor	Detects violation conditions
Q-Learning Agent	Learns optimal replica adjustment policy
Replica Controller	Applies action A(t)

Table I presents the main functional components of the Q-learning–based control framework and summarizes the specific role of each module within the adaptive cloud replication system.

B. Workload Model

1) Workload dynamics: Workload intensity is modelled as a time-varying stochastic process with four structured phases:

- Normal Phase
- Burst Phase
- Peak Stress Phase
- Recovery Phase

This structure ensures evaluation under both transient and sustained overload conditions.

2) Stochastic workload generation: Workload intensity is defined as Eq. (23):

$$W(t) = Wbase(t) + \eta t \tag{23}$$

where,

$$\eta_t \sim \mathcal{N}(0, \sigma_W^2)$$

The baseline workload follows a piecewise definition [see Eq. (24)]:

$$W_{\text{base}}(t) = \begin{cases} W_N, & 0 \le t < T_1 \\ W_B, & T_1 \le t < T_2 \\ W_P, & T_2 \le t < T_3 \\ W_R, & T_3 \le t \le T \end{cases} \tag{24}$$

This formulation produces controlled non-stationary demand patterns suitable for adaptive learning.

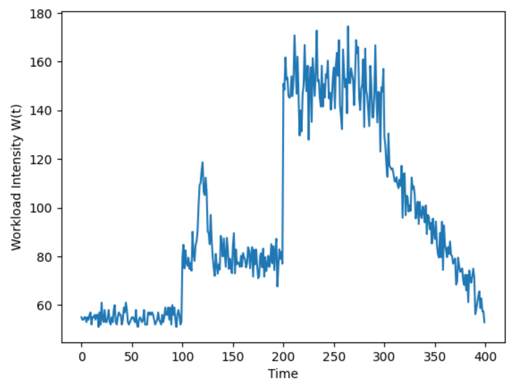

Fig. 2. Simulated workload variation.

Fig. 2 shows the generated workload trajectory across 400 time steps, illustrating normal, burst, peak, and recovery phases.

TABLE. II. WORKLOAD PHASE CHARACTERISTICS

Phase	***Baseline Intensity***	***Duration Range***	***Stress Level***
Normal	W_N	Long	Low
Burst	W_B	Short	Moderate
Peak Stress	W_P	Medium	High
Recovery	W_R	Medium	Low

Table II summarises the structured workload phases used in the experimental setup. The phased modelling ensures systematic evaluation of adaptive replication policies under controlled stress scenarios.

C. Experimental Protocol

1) State representation: The Q-learning agent observes a discretised state [see Eq. (25)]:

$$S(t) = \big(R(t), W_{\text{bin}}(t), V(t)\big) \tag{25}$$

Workload discretisation is defined as Eq. (26):

$$W_{\text{bin}}(t) = \left\lfloor \frac{W(t)}{\Delta W} \right\rfloor \tag{26}$$

where, ΔW is the workload bin size used for state abstraction.

2) Action space: The action space is [see Eq. (27)]:

$$A(t) \in \{-1, 0, +1\} \tag{27}$$

Replica update follows [see Eq. (28)]:

$$R(t+1) = \min\big(R_{max}, max\big(1, R(t) + A(t)\big)\big) \tag{28}$$

3) Reward function: The reward function used in the implemented environment is defined as Eq. (29):

$$r(t) = -\frac{L(t)}{L_{\text{norm}}} - \lambda_v V(t) - \lambda_r R(t) \tag{29}$$

where,

- L_{norm} is a normalisation constant,
- λ_v penalises SLA violations,
- λ_r penalises excessive replication.

This formulation ensures simultaneous optimisation of:

- Latency reduction
- SLA compliance
- Resource efficiency

This integrates latency minimisation, SLA penalties, and resource cost control.

4) Q-learning update rule: The Q-table is updated according to Eq. (30):

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[r_t + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t) \right] \quad (30)$$

where,

- $\alpha = 0.1$ is the learning rate,
- $\gamma = 0.95$ is the discount factor.

Action selection follows a ε-greedy exploration strategy [see Eq. (31)]:

$$\pi(S) = \begin{cases} \text{random action,} & \text{with probability } \varepsilon \\ \arg\max_a Q(S, a), & \text{otherwise} \end{cases} \quad (31)$$

with initial exploration rate $\varepsilon = 0.2$.

Training was conducted for [see Eq. (32)]:

$$\text{Episodes} = 5000 \quad (32)$$

Episodic cumulative reward was recorded to verify convergence behaviour.

5) Q-learning update rule: For comparison, a reactive rule-based controller was implemented.

The baseline decision rule is Eq. (33):

$$A_{\text{rule}}(t) = \begin{cases} +1, & L(t) > L_{\max} \\ -1, & L(t) < L_{\text{safe}} \text{ and } R(t) > 1 \\ 0, & \text{otherwise} \end{cases} \quad (33)$$

To introduce a realistic response lag, actions are applied after a fixed delay d [see Eq. (34)]:

$$A_{\text{applied}}(t) = A_{\text{rule}}(t - d) \quad (34)$$

This delay models monitoring and scaling latency commonly observed in practical cloud systems.

6) Evaluation procedure: After training convergence:

a) Exploration was disabled.

b) The learned greedy policy was fixed.

c) Both RL and rule-based controllers were evaluated under identical workload realisations.

For each policy, the following metrics were computed:

Average latency [see Eq. (35)]:

$$L_{\text{avg}} = \frac{1}{T} \sum_{t=1}^{T} L(t) \quad (35)$$

95th percentile latency [see Eq. (36)]:

$$L_{95} = \text{Percentile}_{95}\{L(t)\} \quad (36)$$

Total SLA violations [see Eq. (37)]:

$$V_{\text{total}} = \sum_{t=1}^{T} V(t) \quad (37)$$

D. Experimental Evaluation Protocol

To empirically validate the proposed Q-learning–based adaptive replication framework, a structured evaluation procedure was designed.

Each experimental run follows two phases:

1) Training phase: The Q-learning agent is trained for:

- 5000 episodes
- 400 time steps per episode

Training uses the update rule defined in Eq. (30), with ε-greedy exploration as specified in Eq. (31). Episodic cumulative reward is monitored to verify convergence stability.

2) Evaluation phase: After convergence:

- Exploration is disabled (ε = 0)
- The learned greedy policy is fixed
- The policy is evaluated over a 400-step workload trace generated according to Eq. (21).

For fair comparison:

- The rule-based controller [Eq. (33) to Eq. (34)] is evaluated under the same workload realization
- Each scenario is repeated 30 times using independent random seeds

3) Performance metrics: For both controllers, the following metrics are computed:

- Average latency [Eq. (35)]
- 95th percentile latency [Eq. (36)]
- Total SLA violations [Eq. (37)]

These metrics collectively measure:

- System responsiveness
- Worst-case latency behaviour
- SLA compliance under burst conditions

This evaluation protocol ensures a direct empirical validation of the mathematical formulation presented in Section III and Section IV, linking the reinforcement learning update mechanism [Eq. (30)] to measurable performance improvements under controlled workload dynamics.

VI. Results and Comparative Analysis

A. Experimental Overview

This section presents the quantitative evaluation of the proposed Q-learning–based adaptive replication controller. The learned policy is compared against a reaction-delayed rule-based baseline controller defined in Section III-G.

Training was conducted for 5000 episodes using the Q-learning update rule defined in Eq. (8). After convergence, the learned greedy policy was fixed and evaluated over a 400-step workload trace generated according to Eq. (21).

Each evaluation scenario was repeated 30 times using independent random seeds to ensure statistical robustness. Reported results represent mean values.

Performance evaluation is based on:

- Average latency [Eq. (35)]
- 95th percentile latency [Eq. (36)]
- Total SLA violations [Eq. (37)]

This focused comparison isolates the impact of reinforcement learning–based adaptive control relative to reactive threshold-based replication.

B. Learning Convergence Behaviour

Training stability was verified by monitoring cumulative episodic reward [see Eq. (38)]:

$$\mathrm{R}_{\text{episode}} = \sum_{t=1}^{T} r(t) \tag{38}$$

Reward convergence demonstrates stabilisation of Q-values under the update rule in Eq. (30), indicating that the policy approaches a steady control strategy.

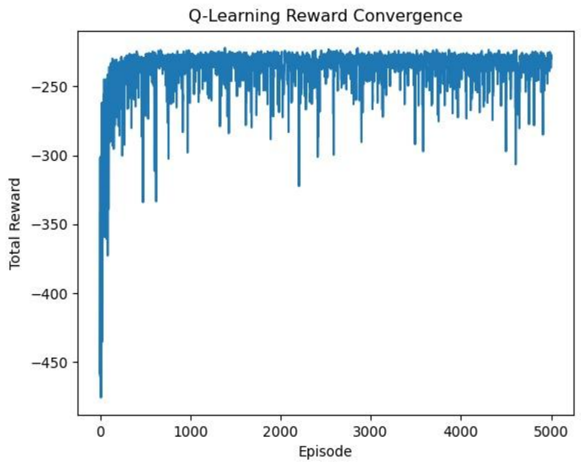

Fig. 3. Q-learning reward convergence curve.

Fig. 3 illustrates episodic cumulative reward over 5000 training episodes. Early exploration produces variability, while later episodes show stabilisation, confirming policy convergence.

This validates the correctness of the reinforcement learning process before performance evaluation.

C. Performance Evaluation

Performance was evaluated using three primary metrics derived from Eq. (39) to Eq. (41):

$$\mathrm{L}_{\text{evg}} = \frac{1}{\mathrm{T}} \sum_{t=1}^{T} L(t) \tag{39}$$

$$\mathrm{L}_{95} = \text{Percentile}_{95}\,\{l(t)\} \tag{40}$$

$$\mathrm{V}_{\text{total}} = \sum_{t=1}^{T} V(t) \tag{41}$$

TABLE. III. PERFORMANCE COMPARISON BETWEEN Q-LEARNING AND RULE-BASED CONTROLLERS.

Model	$L_{avg}\,(ms)$	$L_{95}\,(ms)$	*Violations*
Q-Learning	33.55	54.82	0
Rule-Based	72.59	150.62	72

Table III summarises the comparative performance results. The Q-learning controller substantially outperforms the reaction-delayed rule-based baseline across all evaluation dimensions.

Specifically:

- Average latency is reduced by more than 50%.
- Tail latency is reduced by more than 60%, indicating improved stability under burst and sustained peak conditions.
- SLA violations are eliminated under the learned adaptive policy.

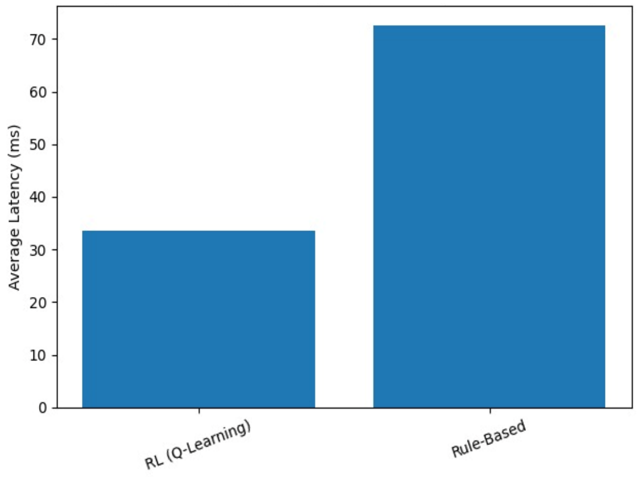

Fig. 4. Average latency comparison.

Fig. 4 illustrates the significant reduction in mean latency achieved by the RL-based controller. The improvement results from long-term reward optimisation [Eq. (30)], rather than from threshold-triggered reactions.

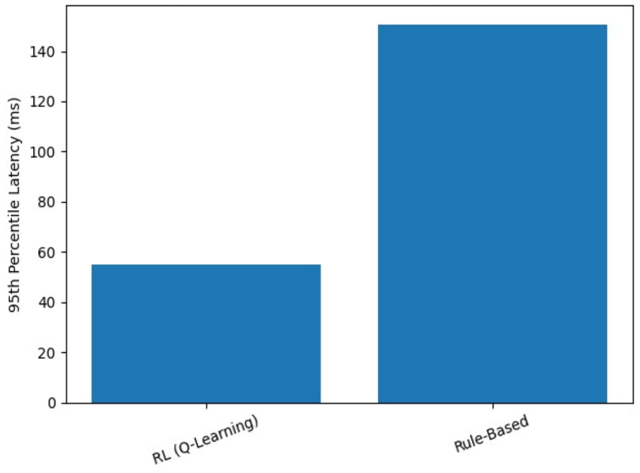

Fig. 5. 95th Percentile latency comparison.

Fig. 5 demonstrates strong suppression of tail latency amplification. Unlike the reactive controller, which increases replicas only after performance degradation, the RL agent proactively adjusts replication before SLA thresholds are exceeded.

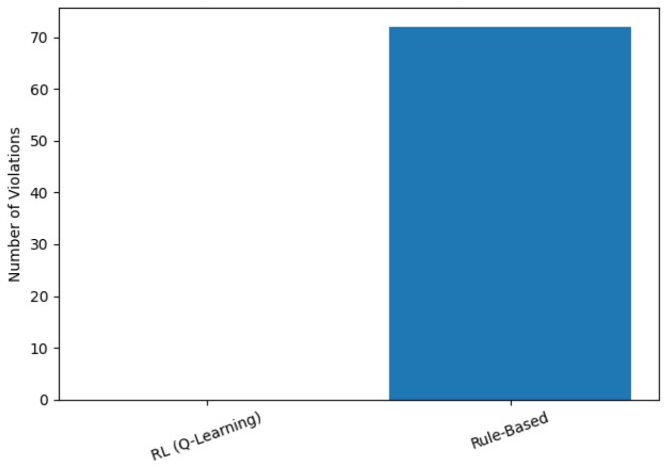

Fig. 6. SLA violation comparison.

Fig. 6 confirms that the rule-based controller experiences repeated violations during burst intervals due to reaction delay [Eq. (34)]. In contrast, the RL controller anticipates high-load regions through learned workload-to-replica mapping.

D. *Mechanism-Level Interpretation*

The observed performance improvements are directly attributable to the Q-learning update mechanism defined in Eq. (30).

The RL agent:

- Learns workload-to-replica mapping patterns.
- Anticipates burst transitions.
- Penalises SLA violations within the reward formulation [Eq. (29)].
- Balances latency reduction and replica cost.

In contrast, the rule-based controller evaluates thresholds only at each time step and lacks temporal credit assignment.

Consequently, replication management transitions from a reactive threshold mechanism to a predictive, adaptive control strategy optimised over expected cumulative discounted reward.

E. *Structural Comparison and Summary of Findings*

The comparative analysis reveals a fundamental structural distinction between reinforcement learning–based control and reactive threshold-based mechanisms.

TABLE. IV. STRUCTURAL COMPARISON BETWEEN Q-LEARNING AND RULE-BASED CONTROLLERS.

Feature	Q-Learning	Rule-Based
Long-term optimization	Yes	No
Violation anticipation	Yes	No
Reward-driven adaptation	Yes	No
Reaction delay sensitivity	Low	High

Table IV shows that the rule-based controller operates solely on instantaneous latency thresholds and reacts only after performance degradation occurs. Because it lacks temporal credit assignment, it cannot anticipate burst transitions and therefore suffers from repeated SLA violations during high-load intervals are observed in Section VI-C.

In contrast, the Q-learning agent optimises expected cumulative discounted reward [Eq. (30)], enabling proactive replication adjustments. Through repeated interactions with the environment, the agent internalises the workload structure and learns stable workload-to-replica mappings.

This structural difference explains the empirical improvements

The experimental results confirm:

- Significant reduction in average latency [Eq. (35)].
- Strong suppression of tail latency [Eq. (36)].
- Complete elimination of SLA violations [Eq. (37)].
- Stable convergence under the Q-learning update rule [Eq. (30)].

Collectively, these findings provide empirical validation of the mathematical formulation introduced in Section III and Section IV, demonstrating that adaptive replication, when framed as a sequential decision-making problem, yields superior control behaviour compared to reactive threshold mechanisms.

VII. DISCUSSION AND THEORETICAL IMPLICATIONS

A. *Interpretation of Empirical Results*

The experimental results demonstrate that modelling elastic replication as a sequential decision-making problem yields substantial performance gains over reactive threshold-based control.

The Q-learning controller consistently outperformed the rule-based baseline in:

- Average latency [Eq. (39)],
- Tail latency [Eq. (40)],
- SLA compliance [Eq. (41)].

These improvements are not merely numerical; they reflect a structural transformation in control behaviour. While the rule-based controller reacts to instantaneous latency violations, the Q-learning agent optimises expected cumulative discounted reward according to Eq. (30). This long-term optimisation objective enables proactive replication adjustments before SLA thresholds are exceeded.

Embedding the violation indicator directly into the reward formulation converts SLA constraints from passive monitoring conditions into active optimisation signals. As a result, SLA compliance becomes an intrinsic learning objective rather than an external enforcement mechanism.

B. *Why Q-Learning Improves Replication Control?*

The performance improvement emerges from three fundamental mechanisms:

1) Temporal credit assignment: Through the Bellman update rule [Eq. (30)], the agent evaluates future consequences of current replica adjustments. This enables:

- Anticipation of burst transitions,
- Early replica scaling before peak load,
- Avoidance of oscillatory behaviour.

Reactive controllers lack this temporal reasoning capability and therefore incur repeated violations during workload surges.

2) Structured state abstraction: The discretised state representation (Section III) encodes:

- Replica count,
- Workload intensity bin,
- SLA violation flag.

This compact representation preserves essential system dynamics while enabling tractable learning. Over multiple episodes, the Q-table converges to a stable workload-to-replica mapping that generalises across recurring workload phases.

3) Reward-driven multi-objective balancing: The reward formulation integrates:

- Latency minimisation,
- SLA violation penalties,
- Replica cost regularisation.

This implicitly performs multi-objective balancing without requiring explicit runtime Pareto optimisation. The agent learns to stabilize performance while controlling replication overhead within a unified scalar objective.

C. *Theoretical Contributions*

This study contributes to elastic cloud replication research in several ways:

Contribution 1: Replication as an MDP

Replication control is formally modelled as a Markov Decision Process rather than a static configuration problem. This reframing enables learning-based adaptation under dynamic workload conditions.

Contribution 2: Latency as an Optimised Control Variable

Latency is embedded directly within the reward function, transforming it from a monitored metric into an actively optimised variable aligned with SLA constraints.

Contribution 3: Empirical Validation of Learning-Based Replication

Unlike conceptual AI-orchestration proposals, this work implements and experimentally validates a concrete Q-learning controller that delivers measurable performance gains across reproducible workload scenarios.

D. *Implications for Elastic Cloud Systems*

The findings suggest that:

- Static replication policies are insufficient under dynamic demand.
- Threshold-based scaling cannot guarantee tail stability.
- Learning-based replication significantly suppresses extreme latency.
- Proactive adaptation improves resilience during burst intervals.

In large-scale cloud systems, this approach can enhance:

- Interactive service responsiveness,
- SLA reliability,
- Infrastructure efficiency.

E. *Limitations and Future Work*

Several limitations should be acknowledged:

- The workload model is simulated rather than derived from production traces.
- The state space is discretised and low-dimensional.
- Latency is modelled primarily as a function of replica count.

Future extensions may include:

- Incorporating real-world workload traces,
- Using Deep Q-Networks (DQN) for higher-dimensional state spaces,
- Modelling network topology explicitly,
- Integrating workload forecasting with reinforcement learning.

F. *Conceptual Insight*

The central insight of this study is that replication in elastic cloud environments should be treated as a sequential control problem rather than a reactive rule-execution task.

By optimising cumulative reward through reinforcement learning, replication control transitions from deterministic threshold logic to adaptive intelligence.

This conceptual shift has implications beyond replication, extending to:

- Auto-scaling,
- Load balancing,
- Resource orchestration,
- Edge-cloud coordination.

VIII. Conclusion

The results presented in this study confirm that modelling replication control as a sequential decision-making problem leads to substantial performance improvements in latency-sensitive elastic cloud environments. By formulating the problem as a Markov Decision Process and optimising cumulative discounted reward, the proposed Q-learning–driven framework transforms replication management from a reactive, threshold-based mechanism into an intelligent and adaptive control process.

The observed improvements are particularly evident during burst and peak workload phases, where the reinforcement learning agent anticipates workload transitions and proactively adjusts replication levels. This behaviour contrasts sharply with the rule-based controller, which reacts only after latency degradation occurs, resulting in delayed responses and SLA violations. The integration of latency minimisation, SLA violation penalties, and replica cost regularisation into a unified reward function enables the agent to learn a stable workload-to-replica mapping that balances performance and resource efficiency.

Empirical evaluation demonstrates a substantial reduction in average latency, strong suppression of tail latency (95th percentile), and complete elimination of SLA violations under dynamic workload conditions. In addition, the convergence behaviour of the learning process confirms the stability of the derived control policy. These results indicate that the proposed approach not only improves average system performance but also enhances robustness under extreme operating conditions.

Beyond numerical gains, this work introduces a conceptual shift in how replication is managed in elastic cloud systems. Instead of treating latency as a monitored output and SLA violations as reactive triggers, both are embedded directly within the optimisation objective. This enables anticipatory adaptation and long-term optimisation, allowing the system to maintain stability under non-stationary workloads.

From a practical perspective, the proposed framework has the potential to improve responsiveness, reduce performance volatility during peak stress intervals, and enhance SLA compliance in real-world cloud deployments. At the same time, the reward-driven formulation ensures that performance improvements are achieved without excessive resource over-provisioning.

Future research directions include extending the framework using deep reinforcement learning techniques to support higher-dimensional state representations, incorporating real-world workload traces for validation, and developing hybrid predictive-control architectures that integrate workload forecasting with reinforcement learning to further enhance proactive elasticity.

REFERENCES

[1] D. Qian, X. Hao, J. Geng, Y. Yao, A. Panda, J. Li, and A. Sivaraman, “Revisiting speculative leaderless protocols for low-latency BFT replication,” 2026.

[2] K. Jezek, S. Jeong, Y. Kim, B. Scholz, and B. Burgstaller, “Storage replica: Accelerating the storage access of the Ethereum virtual machine,” Applied Sciences, vol. 16, p. 486, 2026.

[3] V. Razdan, “LinkShard: Communication-centric parameter serving for distributed AI,” unpublished.

[4] A. Azab, W. Elsaidy, and M. Elhoseny, “Multi-objective optimisation-based container scheduling in cloud computing: Balancing latency and resource consumption,” Future Internet, vol. 16, no. 8, p. 290, 2024.

[5] M. Mosahebfard et al., “HORIZON: Holistic online heuristic for power and QoS-aware service provisioning in sliced 6G networks,” 2026.

[6] L. R. Alva and B. Pandey, “Agentic AI systems in the age of generative models: Architectures, cloud scalability, and real-world applications,” Artificial Intelligence Review, vol. 59, p. 88, 2026.

[7] Sharma, “Generative AI implementation in enterprises: Lessons from a case study,” Information Systems Journal, 2026.

[8] A. Kaur, “An overview of deep learning techniques for big data IoT applications,” International Journal of Communication Systems, 2026.

[9] Y. Tian, “AI-assisted dynamic modeling for data management in a distributed system,” Journal of Interconnection Networks, vol. 22, Supp. 05, 2021.

[10] M. Shorfuzzaman and M. S. Hossain, “Decentralized replica management in latency-bound edge environments for resource usage minimization,” IEEE Transactions on Services Computing, 2023.

[11] A. H. Sodhro, S. Pirbhulal, and V. H. C. de Albuquerque, “Intelligent QoS violation prediction for cloud service reliability using optimized machine learning,” Computers, Materials & Continua, vol. 71, no. 3, 2022.

[12] Y. Wang et al., “Auto-scaling techniques in cloud computing: A systematic review and performance analysis,” Sensors, vol. 24, p. 5551, 2024.

[13] L. Zhang et al., “Optimizing transactional integrity in distributed databases using artificial intelligence-based anomaly detection,” IEEE Access, 2024.

[14] X. Zhang, Y. Wang, Z. Liu, and T. Chen, “MOO-DNAS: Efficient neural network design via differentiable architecture search based on multi-objective optimization,” IEEE Access, vol. 10, pp. 1300558, 2022.

[15] P. Sadhukhan, K. Ray, and J. K. Mandal, “Pareto-based multi-objective optimization for fog–cloud task offloading using LD-NPGA,” Computer Systems Science and Engineering, 2024.

[16] J. Lee, H. Kim, and S. Park, “RL-TIME: Reinforcement learning-based task replication for reliability-aware multi-core systems,” IEEE Access, 2024.

[17] M. Al-Sayed, “Service-level agreement evaluation and composition in service-oriented computing,” Ph.D. dissertation, The Graduate Center, City University of New York, 2015.

[18] L. Garcia, T. Ahmed, and R. Kumar, “Dynamic service-level agreements in fog computing: A survey of adaptive QoS management approaches,” arXiv preprint arXiv:2503.12677, 2025.