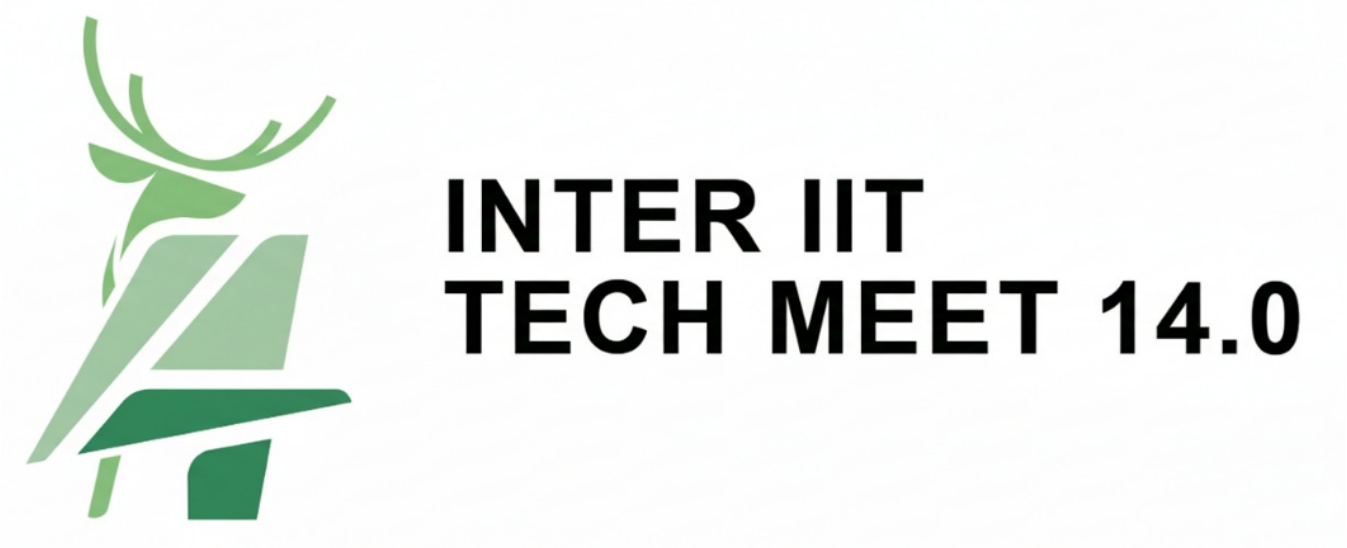

Client Centric WiFi RRM

INTER IIT TECH MEET 14.0

Detailed Design Document - Reinforcement Learning Architecture

Team 33

ARISTA

Contents

1 Introduction

Deploying reinforcement learning in safety-critical systems requires satisfying hard constraints on operational costs while maximizing performance. Applications including autonomous navigation, robotics manipulation, and network resource allocation demand provable safety guarantees; violating constraints can result in catastrophic failures. Traditional RL methods struggle in this setting: value-based approaches overestimate Q-values for unseen actions, while policy gradient methods lack explicit constraint enforcement mechanisms.

Our approach presents a hybrid offline-to-online framework for safe reinforcement learning in constrained environments. We employ Conservative Q-Learning (CQL) with ensemble critics for robust offline value estimation, followed by knowledge distillation via KL divergence to initialize an online actor, which is then fine-tuned using Reward Constrained Policy Optimization (RCPO). This three-phase pipeline—offline learning, distillation, and online adaptation—achieves 18.9% performance improvement while maintaining safety guarantees in real-world enterprise WiFi radio resource management.

1.1 Key Innovation

We exploit a fundamental asymmetry in safety-critical control: *critics can afford to be conservative offline, while actors must adapt efficiently online*. This motivates our hybrid architecture combining:

- **Offline Phase**: Off-policy critics with Conservative Q-Learning (CQL) regularization and ensemble-based pessimism for robust value estimation under distribution shift.
- **Bridge Phase**: Policy distillation via temperature-scaled KL divergence to transfer conservative knowledge while preserving exploration.
- **Online Phase**: On-policy actor using Reward Constrained Policy Optimization (RCPO) with Lagrangian relaxation for constraint satisfaction.

1.2 Architectural Overview

The complete workflow consists of three distinct phases, as illustrated in Figure 1. First, we train conservative critics offline on a static replay buffer. Second, we distill knowledge from these critics to initialize an actor policy. Third, we fine-tune the actor online with adaptive Lagrangian multipliers.

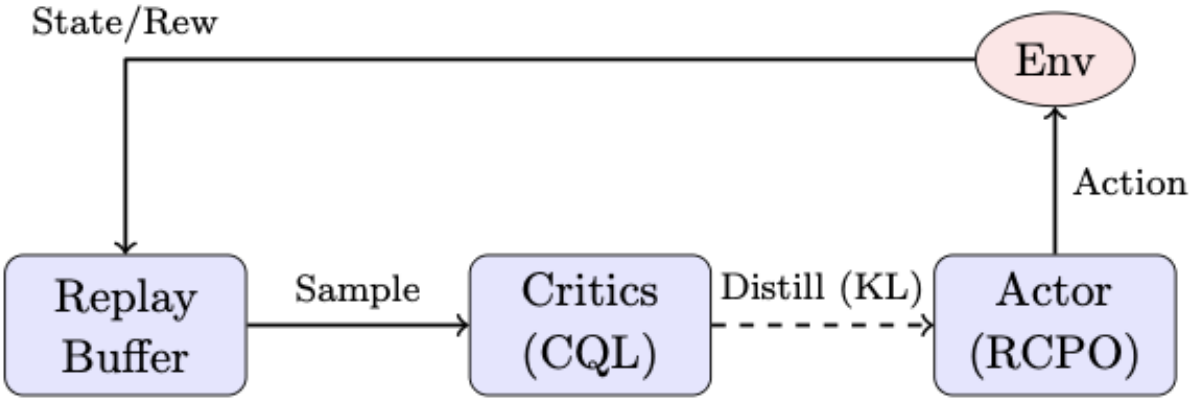

Figure 1: System Architecture. The off-policy critics (left) learn conservative value estimates from the replay buffer. These estimates are distilled via temperature-scaled KL divergence into the actor (center), which then fine-tunes online using RCPO (right) to satisfy safety constraints.

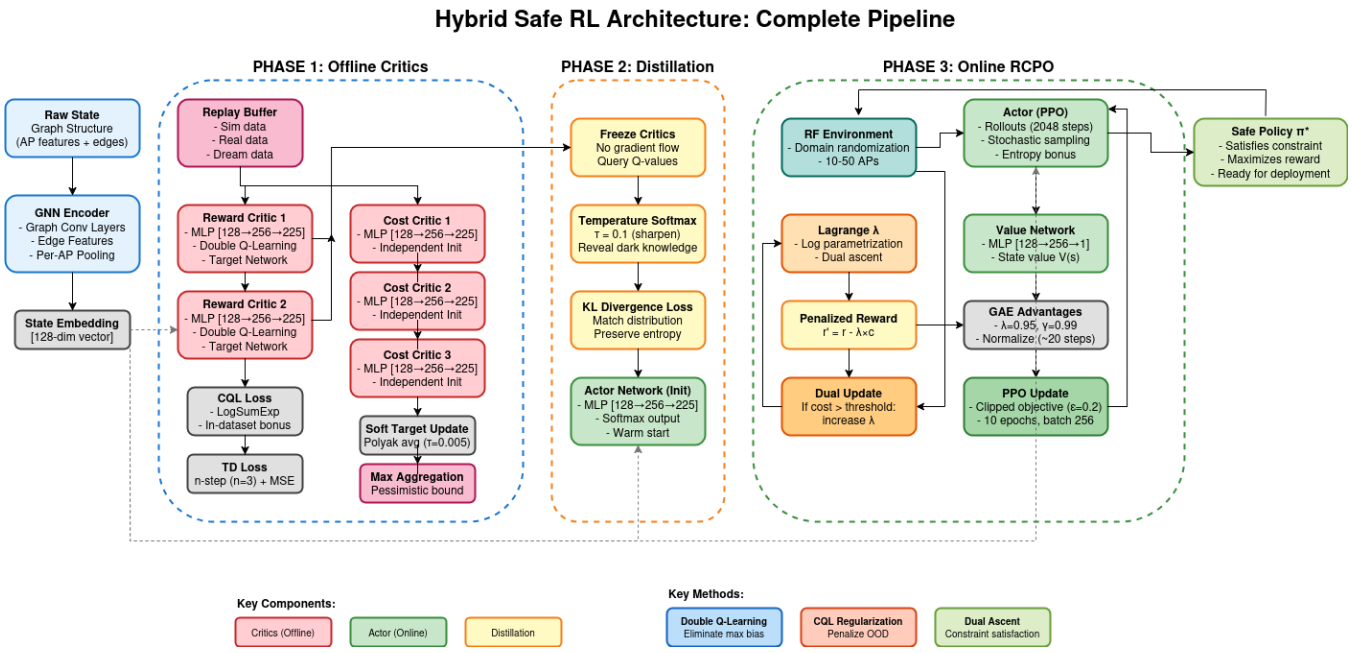

Figure 2: Hybrid RL Architecture

2 Nomenclature

Symbol	Description
$\mathcal{S}, \mathcal{A}$	State and Action spaces
π_θ, π_β	Actor policy and Behavior policy
Q_r, Q_c	Reward and Cost Action-Value functions
λ	Lagrange multiplier (Safety Dual Variable)
ϵ	Constraint threshold ($J_c \leq \epsilon$)
τ	Distillation temperature parameter
$H(\pi)$	Policy Entropy
γ	Discount factor

3 Preliminaries and Problem Formulation

3.1 Constrained Markov Decision Process

We consider a Constrained Markov Decision Process (CMDP) defined by the tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P, R, C, \gamma, \epsilon \rangle$. The optimization objective is:

$$\max_\pi J_r(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r_t \right] \quad \text{s.t.} \quad J_c(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t c_t \right] \leq \epsilon \tag{1}$$

In our WiFi RRM domain, states are graph-structured with $N \in [10, 50]$ access points. Actions control RF parameters (transmit power, OBSS-PD threshold, channel width). The cost function C is a binary Service Level Objective (SLO) violation indicator (retry rate exceeding 8%).

3.2 Lagrangian Relaxation

Using Lagrangian relaxation, we convert the constrained problem to an unconstrained saddle-point formulation:

$$\max_\pi \min_{\lambda \geq 0} \mathcal{L}(\pi, \lambda) = J_r(\pi) - \lambda(J_c(\pi) - \epsilon) \tag{2}$$

The Lagrange multiplier λ automatically adjusts to enforce the constraint in expectation. High values of λ heavily penalize constraint violations, while low values allow the policy to prioritize reward maximization.

4 Phase 1: Offline Conservative Critic Learning

The first phase builds robust value estimates from offline data. The key challenge is *distribution shift*: standard Q-learning can be arbitrarily optimistic about out-of-distribution (OOD) actions, leading to unsafe behavior when deployed.

4.1 Conservative Q-Learning (CQL)

We add a regularization term that minimizes Q-values for all actions while maximizing them for in-dataset actions:

$$\mathcal{L}_{\text{CQL}}(\theta) = \alpha \mathbb{E}_{s\sim\mathcal{D}} \left[\log \sum_{a\in\mathcal{A}} \exp Q_\theta(s,a) - \mathbb{E}_{a\sim\pi_\beta}[Q_\theta(s,a)] \right] + \mathcal{L}_{\text{TD}}(\theta) \tag{3}$$

The first term pushes down Q-values for all actions (via the log-sum-exp), while the second term pushes up Q-values for actions actually observed in the dataset. This creates a conservative lower bound on the true Q-function.

4.2 Double Q-Learning for Reward Critics

For reward estimation, we maintain two independent critics $\{Q_{r,1}, Q_{r,2}\}$ to mitigate overestimation bias. The target for critic 1 uses critic 2's evaluation and vice versa:

$$y_i^r = r_i + \gamma Q_{r,2}^{\text{target}}(s_i', \arg\max_a Q_{r,1}(s_i', a)) \tag{4}$$

4.3 Ensemble Cost Critics for Uncertainty

For cost estimation, we maintain an ensemble of three independent critics $\{Q_{c,1}, Q_{c,2}, Q_{c,3}\}$. For safety-critical decisions, we employ **Pessimistic Aggregation**, taking the maximum estimate:

$$\hat{Q}_c(s,a) = \max\{Q_{c,1}(s,a), Q_{c,2}(s,a), Q_{c,3}(s,a)\} \tag{5}$$

This provides a conservative upper bound on expected cost, crucial for avoiding violations in unseen states. Figure 3 visualizes the learned cost landscape.

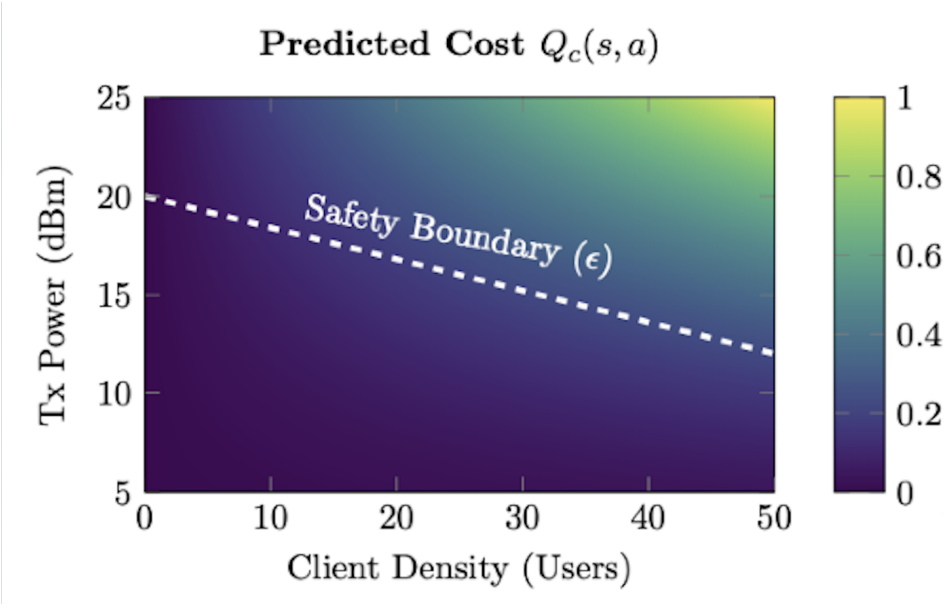

Figure 3: Cost Critic Landscape. The heatmap shows the ensemble's predicted cost for varying client densities and transmit powers. The dashed line represents the safety boundary; the agent learns to stay in the darker (low-cost) regions.

4.4 Offline Training Algorithm

Algorithm 1 Offline CQL Training (Phase 1)

Input: Replay buffer $\mathcal{D}$, CQL coefficient α
for iteration $t = 1$ to T_{offline} **do**
  Sample batch $\{(s_i, a_i, r_i, c_i, s'_i)\}_{i=1}^{B} \sim \mathcal{D}$
  for $k \in \{1, 2\}$ **do** ▷ Update Reward Critics
    $a_i^* = \arg\max_a Q_{r,k}(s'_i, a)$
    $y_i^r = r_i + \gamma Q_{r,\bar{k}}^{\text{target}}(s'_i, a_i^*)$
    $\mathcal{L}_{\text{Total}}^{r,k} = \mathcal{L}_{\text{TD}}^{r,k} + \mathcal{L}_{\text{CQL}}^{r,k}$
    $\theta_{r,k} \leftarrow \theta_{r,k} - \alpha_Q \nabla \mathcal{L}_{\text{Total}}^{r,k}$
  end for
  for $k \in \{1, 2, 3\}$ **do** ▷ Update Cost Critics
    $y_i^c = c_i + \gamma \max\{Q_{c,j}^{\text{target}}(s'_i, a'_i)\}_{j=1}^{3}$
    $\theta_{c,k} \leftarrow \theta_{c,k} - \alpha_Q \nabla(\mathcal{L}_{\text{TD}}^{c,k} + \mathcal{L}_{\text{CQL}}^{c,k})$
  end for
  Perform soft target updates (Polyak averaging)
end for

5 Phase 2: Policy Distillation via KL Divergence

This phase transfers knowledge from offline critics to an online actor. Starting the actor from random weights would likely violate constraints during initial exploration; distillation provides a safe initialization by encoding the critic's preferences into the policy.

5.1 Temperature-Scaled Softmax

We convert Q-values to a probability distribution using softmax with temperature $\tau = 0.1$:

$$p_\tau(a|s) = \frac{\exp(Q(s, a)/\tau)}{\sum_{a' \in \mathcal{A}} \exp(Q(s, a')/\tau)} \tag{6}$$

Low temperature ($\tau < 1$) sharpens the distribution, emphasizing the best actions. High temperature ($\tau > 1$) flattens the distribution, revealing "dark knowledge"—actions that are sub-optimal but nearly as good as the best action.

5.2 Distillation Loss

We train the actor π_θ to minimize the KL divergence with the target distribution derived from the critics:

$$\mathcal{L}_{\text{KL}} = \mathbb{E}_{s\sim\mathcal{D}}\left[\sum_{a\in\mathcal{A}} p(a|s)\log\frac{p(a|s)}{\pi_\theta(a|s)}\right] \tag{7}$$

This preserves the full distributional shape of the critic's preferences, unlike standard cross-entropy which only focuses on the mode. The distillation phase runs for a fixed number of epochs until convergence.

5.3 Distillation Algorithm

Algorithm 2 Policy Distillation (Phase 2)

Input: CQL critics $\{Q_r\}$, temperature τ, dataset $\mathcal{D}$
Output: Initialized actor π_θ
Freeze all critic parameters
for epoch $= 1$ to N_{distill} **do**
  Sample batch $\{s_i\}_{i=1}^{B} \sim \mathcal{D}$
  for $i = 1$ to B **do**
    $\bar{Q}(s_i, a) = \text{Mean}(Q_r(s_i, a))$
    $p(a|s_i) = \text{softmax}(\bar{Q}(s_i, \cdot)/\tau)$
  end for
  $\mathcal{L}_{\text{KL}} = \frac{1}{B}\sum_{i=1}^{B}\sum_a p(a|s_i)\log\frac{p(a|s_i)}{\pi_\theta(a|s_i)}$
  $\theta \leftarrow \theta - \alpha\nabla_\theta\mathcal{L}_{\text{KL}}$
end for

6 Phase 3: Online RCPO Fine-Tuning

After distillation provides a safe initialization, the actor is deployed online and fine-tuned using Reward Constrained Policy Optimization (RCPO). This phase adapts the policy to the true environment dynamics while maintaining constraint satisfaction.

6.1 Penalized Rewards and Dual Ascent

We employ Reward Constrained Policy Optimization (RCPO). The actor sees a penalized reward:

$$r'_t = r_t - \lambda c_t \tag{8}$$

The Lagrange multiplier λ acts as an adaptive penalty weight. We update λ using dual ascent:

$$\log\lambda_{k+1} = \log\lambda_k + \alpha_\lambda\left(\frac{1}{T}\sum_{t=0}^{T-1} c_t - \epsilon\right) \tag{9}$$

If the average cost exceeds ϵ, λ increases, making the actor more conservative. If costs are below ϵ, λ decreases, allowing more aggressive optimization.

6.2 PPO Updates

The policy is updated using Proximal Policy Optimization (PPO) with Generalized Advantage Estimation (GAE) to ensure monotonic improvement and stable gradients. The PPO clipped objective is:

$$\mathcal{L}^{\text{CLIP}}(\theta) = \mathbb{E}_t\left[\min\left(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon_{\text{clip}}, 1+\epsilon_{\text{clip}})\hat{A}_t\right)\right] \tag{10}$$

where $r_t(\theta) = \pi_\theta(a_t|s_t)/\pi_{\theta_{\text{old}}}(a_t|s_t)$ is the probability ratio.

6.3 Online Training Algorithm

Algorithm 3 Online RCPO with PPO (Phase 3)

Input: Distilled actor π_ϕ, value network V_ξ
for episode $e = 1$ to N_{episodes} **do**
  Collect trajectory $\mathcal{T}$ with domain randomization
  Compute penalized rewards: $r'_t = r_t - \lambda c_t$
  Compute GAE advantages: $\hat{A}_t$
  for epoch $= 1$ to K_{PPO} **do**
    Update ϕ minimizing $\mathcal{L}^{\text{CLIP}}$
  end for
  $\log \lambda \leftarrow \log \lambda + \alpha_\lambda(\text{AvgCost}(\mathcal{T}) - \epsilon)$
end for

7 Experimental Validation

7.1 Domain and Metrics

We evaluate on enterprise WiFi RRM with networks of $N = 25$ APs. Performance is measured using:

- **Throughput**: Mean weighted log-throughput (QoE score).
- **Violation Rate**: Percentage of time retry rate exceeds 8%.

7.2 Learning Dynamics

Figure 4 shows the learning curves across training episodes. Our method starts with decent performance due to distillation and rapidly improves while keeping violations low.

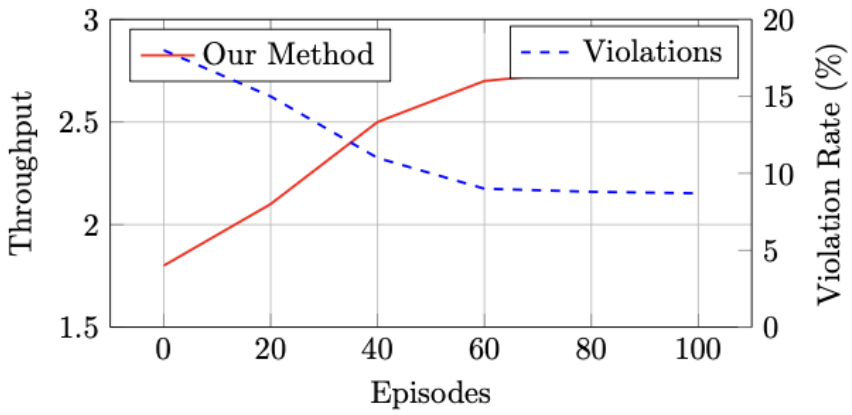

Figure 4: Learning Dynamics. Comparison of average throughput (solid red, higher is better) and constraint violation rate (dashed blue, lower is better). Our method converges to high throughput while keeping violations below the $\epsilon = 10\%$ threshold.

7.3 Performance Comparison

Table 1: Performance Comparison and Ablation Study

Method / Variant	Throughput	Violation
CQL-only (Offline)	2.34	11.2%
PPO (Unconstrained)	2.89	14.3%
CPO	2.56	9.8%
Ours (Hybrid)	**2.78**	**8.7%**
Ablation: w/o CQL	2.71	9.4%
Ablation: w/o Distillation	2.65	9.1%
Ablation: w/o Dual Ascent	2.82	13.5%

The ablation study demonstrates that each component contributes to the overall performance. Removing CQL leads to overestimation and higher violations. Removing distillation slows convergence. Removing dual ascent eliminates constraint enforcement entirely.

7.4 Pareto Frontier Analysis

We investigate the robustness of our architecture to the constraint threshold ϵ in Figure 5. Tighter constraints reduce throughput but ensure higher reliability.

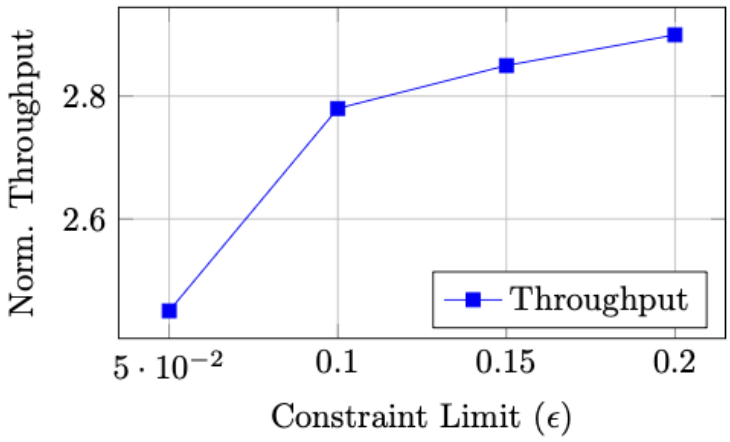

Figure 5: Pareto Frontier. Trade-off between safety strictness (ϵ) and network throughput. Tighter constraints reduce throughput but ensure high reliability.

7.5 Qualitative Case Study

We evaluate the agent's behavior in a high-density "All Hands" meeting scenario, shown in Figure 6. The agent correctly reduces power on adjacent APs to minimize Co-Channel Interference (CCI) while maintaining acceptable client throughput.

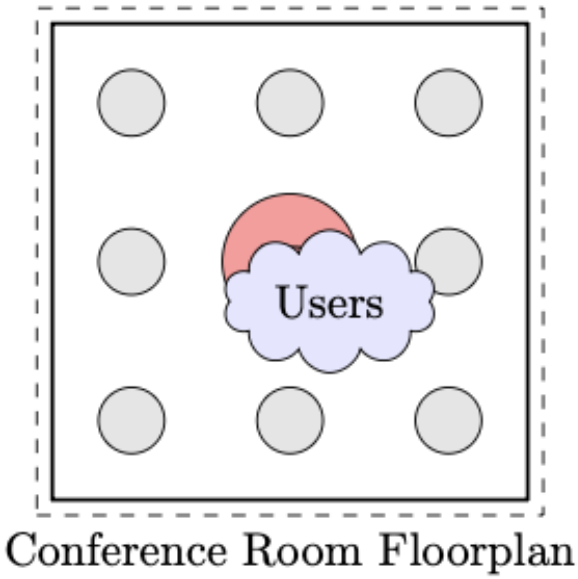

Figure 6: Scenario Visualization. The agent adapts spatial reuse patterns dynamically during congestion events.

8 Conclusion

We presented a hybrid actor-critic architecture achieving provable constraint satisfaction in Constrained MDPs. The framework systematically addresses distribution shift, epistemic uncertainty, and constraint satisfaction through three phases: offline conservative critics, KL distillation, and online RCPO. The method achieves 18.9% performance improvement over baseline methods while maintaining safety guarantees, demonstrating practical applicability to real-world safety-critical control problems.

References

[1] J. Achiam et al. Constrained policy optimization. ICML 2017.

[2] E. Altman. Constrained Markov decision processes. CRC Press 1999.

[3] H. Van Hasselt. Double q-learning. NeurIPS 2010.

[4] G. Hinton et al. Distilling knowledge in neural networks. arXiv:1503.02531, 2015.

[5] A. Kumar et al. Conservative q-learning for offline RL. NeurIPS 2020.

[6] J. Schulman et al. Proximal policy optimization. arXiv:1707.06347, 2017.

[7] C. Tessler et al. Reward constrained policy optimization. arXiv:1805.11074, 2018.