Risk-Aware Constrained Reinforcement Learning with Non-Stationary Policies

Zhaoxing Yang
Shanghai Jiao Tong University
Shanghai, China
yiannis@sjtu.edu.cn

Haiming Jin*
Shanghai Jiao Tong University
Shanghai, China
jinhaiming@sjtu.edu.cn

Yao Tang
Shanghai Jiao Tong University
Shanghai, China
tangyao2020@sjtu.edu.cn

Guiyun Fan*
Shanghai Jiao Tong University
Shanghai, China
fgy726@sjtu.edu.cn

ABSTRACT

Constrained reinforcement learning (RL) algorithms have attracted extensive attentions nowadays to tackle sequential decision-making problems that contain constraints defined under various risk measures. However, most works only search policies within the stationary policy class and fail to capture a simple intuition: adjust the action-selecting distribution at each state according to the accumulated cost so far. In this work, we design a novel quantile-level-driven policy class to fully realize such intuition, within which each policy additionally takes the quantile level of the accumulated cost as input. Such quantile level is obtained via a novel Invertible Backward Distributional Critic (IBDC) framework, which utilizes invertible function approximators to estimate the accumulated cost distribution and outputs the required quantile level with their inverse forms. Further, the estimated accumulated cost distribution also helps to decompose the challenging trajectory-level constraints into state-level constraints, and Risk-Aware Constrained RL (RAC) algorithm is designed then to solve the decomposed problem with Lagrangian multipliers. Experimental results in various environments validate the effectiveness of RAC versus state-of-the-art baselines.

KEYWORDS

Reinforcement Learning, Constrained Markov Decision Process, Risk-Aware Learning

ACM Reference Format:
Zhaoxing Yang, Haiming Jin, Yao Tang, and Guiyun Fan. 2024. Risk-Aware Constrained Reinforcement Learning with Non-Stationary Policies. In *Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 9 pages.

*Corresponding authors.

Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand. © 2024 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

1 INTRODUCTION

Sequential decision-making problems often involve various constraints introduced by physical limitations [37], budget restrictions [7, 9], as well as requirements on certain performance metrics [34]. Constrained Markov Decision Process (CMDP), whose goal is to maximize the expected long-term reward while limiting the expected long-term cost below the constraint threshold, is a natural choice to formulate such problems and has become a research hot spot recently [2, 3, 26].

However, CMDP becomes insufficient for many applications (e.g., autonomous driving and industrial robotics [6, 27]) with safety-critical constraints, as it only constrains the expected long-term cost, and even the optimal policy of a CMDP may still violate the constraints in each execution. Several works have noticed such gap and propose formulations by taking the distribution of the long-term cost into consideration with various risk measures. For instance, [11, 32] utilize the Conditional Value-at-Risk (CVaR_α) measure to limit the expectation of the top α fraction of the long-term cost distribution below the constraint threshold. [15, 27, 29, 31] impose hard constraints in problems and require the cost of any trajectory below the constraint threshold.

Nevertheless, the above formulations may still be sub-optimal, as they fail to capture a simple intuition about the non-stationarity of policies on the accumulated cost. Take the problems with hard constraints as an example. When the agent visits the same state with two different accumulated costs, say one is very close to the threshold and the other is far less than the threshold, the optimal action-selecting distribution at this state should also be distinct. That is, the one with little cost could be more adventurous to collect more reward, while the other should be more conservative to not violate the constraint. However, such distinction is unrealizable in the above formulations as they only consider the stationary policy class, where each policy always generates the same action-selecting distribution at the same state. Recently, such intuition has motivated works [23, 24], where the left budget is recorded and augmented to the state space. However, one major limitation of their algorithms is that they require the risk measure to be hard constraints or risk-neural constraints, and they are inapplicable for other classical risk measures, such as CVaR_α.

To address the limitations discussed above, this paper aims to propose and solve a new problem formulation for safety-critical

applications, which fully captures the above non-stationarity intuition and is applicable for general risk measures. Firstly, a new quantile-level-driven policy class is designed as the space to search policies, within which each policy not only takes the state as input, but also takes the quantile level of the current accumulated cost as input. The quantile level is obtained via a novel Invertible Backward Distributional Critic (IBDC) framework, where invertible function approximators are designed to estimate the accumulated cost distribution by approximating its quantile function, and the inverse forms of such approximators take the current accumulated cost as input and output the corresponding quantile level.

In addition to providing information to help decision-making, the IBDC framework is also essential for decomposing the original trajectory-level constraints imposed only on the initial state into state-level constraints imposed on each state along the trajectory due to the estimated accumulated cost distribution. Such decomposition is necessary to obtain better solutions as the former one constrains the cost of the full-trajectory but provides no information on how to and how much to update policy at each state along the trajectory. Finally, we propose the Risk-Aware Constrained RL (RAC) algorithm to solve the decomposed problem with Lagrangian multipliers. Our main contributions are as follows.

- We propose a novel quantile-level-driven policy class for constrained RL problems, and demonstrate its advantages versus the common stationary policy class with an intuitive example. A novel IBDC framework is designed then to estimate the accumulated cost distribution and obtain the required quantile level in the policy class.
- With the estimated accumulated cost distribution, we also decompose the challenging trajectory-level constraints into state-level constraints, and desgin the RAC algorithm to solve it by alternatively updating the policy and the Lagrangian multipliers.
- Finally, we visualize the proposed IBDC framework in two environments to show its effectiveness for outputting the quantile levels to help decision-making. Later, experiments in safety gym benchmarks and simulations of two real-world applications demonstrate the superiority of RAC versus state-of-the-art baselines.

2 PROBLEM FORMULATION

We consider the finite-horizon constrained Markov decision process represented by $(\mathcal{T}, \mathcal{S}, \mathcal{A}, p, r, c, s_0)$. $\mathcal{T} = \{0, 1, \cdots, T\}$, where $T \in \mathbb{N}$, contains all the time steps. $\mathcal{S}$ denotes the state space. $s_0 \in \mathcal{S}$ is the initial state, and $\mathcal{A}$ denotes the action space. Given the state s_t and a_t at time step $t \in \mathcal{T}$, $p(s_{t+1}|s_t, a_t)$ gives the transition probability to the next state s_{t+1}. $r(s_t, a_t)$ and $c(s_t, a_t)$ output the deterministic reward and cost, respectively.

The stationary policy class[1] $\Pi_{\text{sta}} = \{\pi|\pi : \mathcal{S} \to \Delta(\mathcal{A})\}$ is the primary policy class considered in previous constrained RL works, where each policy always generates the same action distribution at the same state. However, such policy class may lead to sub-optimal solutions, as shown in Sec. 3 with an intuitive example. Instead, this paper considers the following non-stationary[2] policy class.

[1]We use $\Delta(\cdot)$ to denote the space of all distributions over $\cdot$.

[2]We clarify that, term "non-stationarity" in this work refers that the policies also depend on the quantile level additionally, instead of the state alone. In fact, "non-stationarity" in literatures means the ever-changing environment [10], which is substantially different from ours. Nevertheless, we still use the term to emphasize the policy dependence on the quantile level other than the state alone.

Definition 2.1 (Quantile-level-driven Policy Class)**.** For any trajectory $s_0, a_0, \cdots, s_{t-1}, a_{t-1}, s_t$, policy π samples actions at state s_t according to $\pi(\cdot|x_t)$, where $x_t = (s_t, \tau_t) \in \mathcal{X}$, and $\mathcal{X} = \mathcal{S} \times [0, 1]$. Let $Z(s_t, \pi)$ be the random variable of the accumulated cost distribution before reaching state s_t. Then, $\tau_t \in [0, 1]$ is the quantile level[3] of $Z(s_t, \pi)$ at realization $\sum_{t'=0}^{t'=t-1} c(s_{t'}, a_{t'})$. The set of all such policies $\Pi_{\text{nsta}} = \left\{\pi\middle|\pi : \mathcal{X} \to \Delta(\mathcal{A})\right\}$ is referred to as the quantile-level-driven policy class.

Note that the additional τ_t keeps the definitions of reward and cost function unchanged, and it transits deterministically to τ_{t+1} for any (s_t, a_t) pair given policy π. Hence, we will use $x_t \in \mathcal{X}$ to replace $s_t \in \mathcal{S}$ in reward, cost and transition function with slight notation abuse. For each policy $\pi \in \Pi_{\text{nsta}}$, $R^\pi(x_t, a_t) = \sum_{t'=t}^{T} r(x_{t'}, a_{t'})$ and $C^\pi(x_t, a_t) = \sum_{t'=t}^{T} c(x_{t'}, a_{t'})$ are the random variables of the long-term reward and cost under policy π that start from (x_t, a_t), respectively. Then, the classical action-value function is $Q^\pi(x_t, a_t) = \mathbb{E}R^\pi(x_t, a_t)$. Let the distorted expectation of C^π be $\Phi_{g_c}\left[C^\pi(x_t, a_t)\right] = \int_0^1 F^{-1}_{C^\pi(x_t,a_t)}(\tau) g_c(\tau) d\tau$, where $F^{-1}_{C^\pi(x_t,a_t)}$ is the quantile function of random variable $C^\pi(x_t, a_t)$, and $g_c : [0, 1] \to [0, 1]$ is the distortion risk measure that specifies the weights for quantile levels. Based on these definitions, this paper aims to solve the following Risk-Aware Constrained Optimization (RACO) problem, which finds policy π^* such that

$$\begin{aligned} \pi^* \in \arg\max_{\pi \in \Pi_{\text{nsta}}} & \mathbb{E}_{a_0 \sim \pi(\cdot|x_0)}\left[Q^\pi(x_0, a_0)\right] \\ \text{s.t. } & \mathbb{E}_{a_0 \sim \pi(\cdot|x_0)} \Phi_{g_c}\left[C^\pi(x_0, a_0)\right] \le d, \end{aligned} \tag{1}$$

where d is the constraint threshold, $x_0 = (s_0, 1)$ is the initial state. In problem (1), Q^π can be evaluated by applying the classical Bellman operator iteratively [25], while evaluating the distribution of $C^\pi(x_t, a_t)$ requires the distributional Bellman operator $\mathcal{T}^\pi$: $\mathcal{T}^\pi C(x_t, a_t) \overset{D}{=} c(x_t, a_t) + C(x_{t+1}, a_{t+1})$, where $x_{t+1} \sim p(\cdot|x_t, a_t)$, $a_{t+1} \sim \pi(\cdot|x_{t+1})$, and C is initialized from any random distribution. $\overset{D}{=}$ indicates equality in distribution. It has been shown that $\mathcal{T}^\pi$ is a $1 - 1/T$-contraction in the Wasserstein distance [5], and C^π is the unique fixed point of $\mathcal{T}^\pi$.

Generality of RACO. RACO takes several classical constrained RL formulations as specical cases. For instance, when $g_c = \text{Uniform}([0, 1])$, $\Phi_{g_c}(C^\pi(s_t, a_t)) = Q_c^\pi(s_t, a_t)$ recovers the classical action-cost function, and RACO reduces to the risk-neutral CMDP formulation [3]. When $g_c = \text{Uniform}([\xi, 1]), \xi \in (0, 1)$, Φ_{g_c} becomes the classical CVaR [19, 20] measure, and RACO reduces to the CVaR-constrained formulation considered in [11, 32]. When g_c is the Dirac delta distribution at 1, RACO constrains the maximal long-term cost among the supports of the distribution of $C^\pi(s_0, a_0)$ to be lower than the threshold d, which in fact reduces to the hard-constraint case [23, 24, 27, 29]. Furthermore, RACO also induces several novel formulations with specific distortion risk measure g_c. For example, RACO is risk-averse when $g_c = \text{Uniform}([\xi, 1])$ and becomes risk-seeking when $g_c = \text{Uniform}([0, \xi])$. The latter one may be attractive in financial applications like portfolio management. Other risk

[3]In this paper, for any random variable Y with quantile function $F_Y^{-1}(\tau) = y$, we define the input probability τ as the *quantile level*, and the output value y as the *quantile*.

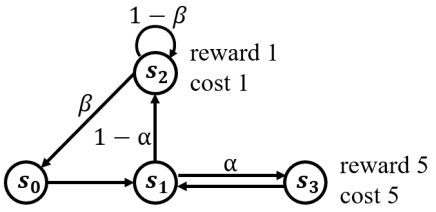


Figure 1: HIHR environment.

measures compatible here include cumulative probability weighting (CPW) [28] that defines $g_c(\tau) = \tau^\beta/(\tau^\beta + (1-\tau)^\beta)^{\frac{1}{\beta}}$; Wang [30] that has $g_c(\tau) = F_{\mathcal{N}}(F_{\mathcal{N}}^{-1}(\tau) + \beta)$, where β is the hyper-parameter and $F_{\mathcal{N}}$ is the Cumulative Distribution Function (CDF) for standard Gaussian distribution.

3 LIMITATIONS OF STATIONARY POLICIES

We illustrate the limitations of the stationary policy class Π_{sta} by an intuitive High Income High Risk (HIHR) environment, as shown in Fig. 1. In HIHR, s_0 is the initial state, s_1, s_2 have two available actions, and s_2, s_3 have non-zero reward and cost. HIHR has total horizon 10. To further simplify the computation, β is set to be 10^{-6} by default, and $\alpha \in [0, 1]$ becomes the only decision variable.

Intuitively, an agent will get the maximal expected reward 25 and cost 25 in the unconstrained setting by letting $\alpha = 1$. In the constrained settings, we set the threshold to be 15, $g_c = \text{Uniform}([\xi, 1.0])$ with $\xi \in \{0.0, 0.5, 0.9\}$. We solve the instantiated RACO problems with brute-force enumeration of α. Note that each policy in the stationary policy class Π_{sta} can only set one value of α, while for each policy in Π_{nsta}, it could set different values of α every time visiting state s_1 along the trajectory due to inputting different quantile levels. The approximately maximal expected return among feasible policies in the stationary policy class Π_{sta} is 15.00, 12.07, 9.40, respectively, and 15.00, 14.40, 13.93 in Π_{nsta}. The results suggest that the non-stationary policy class Π_{nsta} always has a better policy than the stationary policy class Π_{sta}, as policies within Π_{nsta} could "adjust" the action-selecting distribution at s_1 timely to become adventurous on reward or conservative for constraint-satisfaction according to the level of the current accumulated cost. More results including the solved values of α are shown in Appendix B.1 due to space limits.

4 INVERTIBLE BACKWARD DISTRIBUTIONAL CRITICS

One major challenge for solving problem (1) is that Π_{nsta} requires the quantile level of the accumulated cost. In this section, we tackle such challenge with a novel IBDC framework. Moreover, the accumulated cost distribution estimated by IBDC is also essential for the decomposed problem shown in Sec. 5.

4.1 Backward Markov Chain and Backward Cost Distribution

We firstly make the following mild assumption to guarantee the existence of unique stationary state distribution, which is widely adopted in literatures [18, 21].

Assumption 4.1. For any policy $\pi \in \Pi_{\text{nsta}}$, we assume the Markov chain induced by transition $p^\pi(x_{t+1}|x_t) = \sum_{a\in\mathcal{A}} p(x_{t+1}|x_t, a)\pi(a|x_t)$ to be irreducible and aperiodic.

Let the unique stationary state distribution be μ^π, then $\mu^\pi(x_{t+1}) = \sum_{x_t\in\mathcal{X}} p^\pi(x_{t+1}|x_t)\mu^\pi(x_t)$ by definition. From Bayesian's rule, we note the following probability

$$\overleftarrow{p}^\pi(x_t|x_{t+1}) = \frac{\sum_{a\in\mathcal{A}} p(x_{t+1}|x_t, a)\pi(a|x_t)\mu^\pi(x_t)}{\mu^\pi(x_{t+1})}$$

characterizes the probability that a previous state x_t has been visited to reach the current state x_{t+1}. The Markov chain characterized by $\overleftarrow{p}^\pi$ is referred to as the *backward Markov chain*.

With the backward Markov chain, the random variable of the *backward cost distribution*[4] at some state x_t is defined as $\overleftarrow{C}^\pi(x_t, a_t) = \sum_{k=0}^{T_B} c(x_{t-k}, a_{t-k})$, where $a_{t-k} \sim \pi(\cdot|x_{t-k}), x_{t-k-1} \sim \overleftarrow{p}^\pi(\cdot|x_{t-k})$, and T_B is the steps it takes for reaching state x_0. Note that T_B is well-defined as [18, 21] have shown that the backward Markov chain is also ergodic with Assump. 4.1. Thus, each policy π will reach the initial state x_0 (i.e., the terminal state in the backward Markov chain) in finite steps.

However, estimating $\overleftarrow{C}^\pi(x_t, a_t)$ by definition is intractable as $\overleftarrow{p}^\pi$ is unknown. To mitigate this, we firstly show that $\overleftarrow{C}^\pi(x_t, a_t)$ is equivalent in distribution to the random variable of the accumulated cost from a forward view (Lemma 4.2), and then complete the estimation with a novel backward distributional Bellman operator (Lemma 4.3).

Lemma 4.2. $\overleftarrow{C}_k^\pi(x_t, a_t) \overset{D}{=} C_k^\pi(x_{t-k}, a_{t-k}), \forall k \in \mathbb{N}$.

In the lemma, $\overleftarrow{C}_k^\pi(x_t, a_t) = \sum_{k'=0}^{k} c(x_{t-k'}, a_{t-k'})$ is the random variable of the distribution of the accumulated cost in the last k steps to x_t from a backward view, where $x_{t-k'} \sim \overleftarrow{p}^\pi(\cdot|x_{t-k'+1})$ and $a_{t-k'} \sim \pi(\cdot|x_{t-k'})$. $C_k^\pi(x_{t-k}, a_{t-k}) = \sum_{k'=0}^{k} c(x_{t-k+k'}, a_{t-k+k'})$ is the random variable of the distribution of the accumulated cost in the future k steps, starting from $x_{t-k} \sim \mu^\pi(\cdot)$, and $x_{t-k+k'} \sim p^\pi(\cdot|x_{t-k+k'-1}), a_{t-k+k'} \sim \pi(\cdot|x_{t-k+k'})$. Proof is given in Appendix A.1 Essentially, Lemma 4.2 reveals that the cumulative cost from x_{t-k} to x_t shares the same distribution from the forward view (i.e., with transition p^π) and the backward view (i.e., with transition $\overleftarrow{p}^\pi$). Hence, one can estimate the distribution of $\overleftarrow{C}^\pi(x_t, a_t)$ unbiasedly with samples from the forward chain.

Moreover, utilizing the recursive property $\overleftarrow{C}^\pi(x_t, a_t) \overset{D}{=} c(x_t, a_t) + \overleftarrow{C}^\pi(x_{t-1}, a_{t-1})$, we define the backward distributional Bellman operator as $\overleftarrow{\mathcal{T}}_c^\pi \overleftarrow{C}(x_t, a_t) \overset{D}{=} c(x_t, a_t) + \overleftarrow{C}(x_{t-1}, a_{t-1})$, where $\overleftarrow{C}$ is drawn from any distribution. By iteratively applying $\overleftarrow{\mathcal{T}}_c^\pi$ over $\overleftarrow{C}$, the distribution of $\overleftarrow{C}$ converges to $\overleftarrow{C}^\pi$ as $\overleftarrow{\mathcal{T}}_c^\pi$ is a contraction mapping. Proof is given in Appendix A.2.

Lemma 4.3. *$\overleftarrow{\mathcal{T}}_c^\pi$ is a contraction mapping in the Wasserstein distance, and $\overleftarrow{C}^\pi$ is the fixed point, i.e., $\overleftarrow{C}^\pi = \overleftarrow{\mathcal{T}}_c^\pi \overleftarrow{C}^\pi$.*

[4] We use term *backward cost distribution* and *accumulated cost distribution* interchangeably in this paper.

4.2 Invertible Backward Distributional Critic

With Lemma 4.2 and 4.3, one can estimate $\overleftarrow{C}^{\pi}$ with popular distributional RL approaches, such as categorical-based DRL [5] and quantile-based DRL [12, 16]. However, these approaches cannot obtain the quantile level that Π_{nsta} needs, as they target at unconstrained problems and they are not designed for such purpose.

To mitigate such challenge, our inspiration comes from the fact that what we need for Π_{nsta} is actually the CDF of the backward cost distribution, and CDF is the inverse of the quantile function. Hence, it becomes straightforward to get the CDF if invertible function approximators are used in the quantile-based DRL [12, 16] approaches to estimate the quantile function of the backward cost distribution. We refer these invertible function approximators as Invertible Backward Distributional Critic (IBDC). In practice, one can initialize any f from an IBDC class, and get f updated with classical losses adopted in quantile-based DRL approaches, such as the Huber quantile regression losses. Meanwhile, the quantile level of any realized accumulated cost C can be obtained by simply inputting C into the inverse form of f. We consider two such IBDC classes in experiments, namely $\mathcal{F}_{linear}$ and $\mathcal{F}_{nn}$.

$$\mathcal{F}_{linear} = \Big\{ f \Big| f(x_t, a_t, \tau_t) = w^T \cdot [s_t, a_t, \tau_t] + b, w \in \mathbb{R}^n, b \in \mathbb{R} \Big\},$$

$$\mathcal{F}_{nn} = \Big\{ f \Big| f(s_t, a_t, \tau_t) = w_1 \mathcal{N}_1(s_t, a_t) \cos(\pi \tau_t) + w_2 \mathcal{N}_2(s_t, a_t), \; w_1 = \mathcal{N}_3(s_t, a_t), w_2 = 1 - w_1 \Big\}.$$

In $\mathcal{F}_{linear}$, linear function approximators are used with state s_t, action a_t and quantile τ_t stacked as the feature vector. n represents the dimension of the stacked vector. With realized cost C, the corresponding quantile level τ_t can be calculated as $\tau_t = \frac{C - b - w_{[1:n-1]}^T \cdot [s_t, a_t]}{w_n}$. In $\mathcal{F}_{nn}$, $\mathcal{N}_1, \mathcal{N}_2 : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, $\mathcal{N}_3 : \mathcal{S} \times \mathcal{A} \to [0, 1]$ are neural networks, and $\mathcal{N}_3$ assigns weights to trade-off the two additive terms. Quantile level of the realized cost C can be calculated by $\tau_t = \frac{1}{\pi} \cdot \arccos \frac{C - w_2 \mathcal{N}_2(s_t, a_t)}{w_1 \mathcal{N}_1(s_t, a_t)}$. In practice, we also add small positive ϵ into denominator and clip operations during the calculation to keep it valid when outputting the quantile level.

5 RAC WITH DECOMPOSED CONSTRAINED POLICY IMPROVEMENT

Another challenge for solving problem (1) is that its constraint is defined on the long-term cost of the full trajectory. Since the cost of the full trajectory is influenced by the action-selecting distributions at each state along the trajectory, problem (1) is hard to solve directly as it provides no information on how to and how much to update policy at each state along the trajectory. In this section, we mitigate such challenge by decomposing the trajectory-level constraint into state-level constraint, and solve the latter one by explicitly control the decisions at each state along the trajectory, given the quantile level of the accumulated cost up to the state.

5.1 Decomposed Constrained Policy Improvement

The decomposition of the trajectory-level constraint relies on the following lemma on the cost quantile function. The proof is inspired by the Frechet-Hoeffding inequality and is given in Appendix A.3.

Lemma 5.1. *For any trajectory $\{(x_i, a_i)\}_{i=0}^T$ generated by $\forall \pi \in \Pi_{nsta}$, we have*

$$F^{-1}_{C^\pi(x_0,a_0)}(\tau) \le F^{-1}_{\overleftarrow{C}^\pi(x_{t-1},a_{t-1})}(\nu) + F^{-1}_{C^\pi(x_t,a_t)}(1 - \nu + \tau),$$

where $t \in [1, \cdots, T]$, $\nu \in [0, 1]$ is the quantile level of random variable $\overleftarrow{C}^\pi(x_{t-1}, a_{t-1})$ at realization $\sum_{t'=0}^{t'=t-1} c(x_{t'}, a_{t'})$, and $\forall \tau \in [0, \nu]$.

Lemma 5.1 shows that, the τ-quantile of random variable $C^\pi(x_0, a_0)$ can be decomposed into two parts. The first part $F^{-1}_{\overleftarrow{C}^\pi(x_{t-1},a_{t-1})}(\nu)$ is the ν-quantile of the backward cost distribution before reaching x_t, and the second part $F^{-1}_{C^\pi(x_t,a_t)}(1 - \nu + \tau)$ is the $(1 - \nu + \tau)$-quantile of the future cost distribution when executing action a_t at state x_t. Thus, if agent is at state x_t right now and has collected cost with quantile level ν, he could control the τ-quantile of the long-term cost distribution of the full trajectory by deciding the future cost at quantile level $(1 - \nu + \tau)$. Such intuition inspires the following Decomposed Constrained Policy Improvement (DCPI) problem: $\forall x_{t-1}, a_{t-1}, x_t$,

$$\begin{aligned} &\pi_{k+1} \in \operatorname*{argmax}_{\pi \in \Pi_{\text{nsta}}} \left\langle \pi(\cdot|x_t), Q^{\pi_k}(x_t, \cdot) \right\rangle \\ &\text{s.t. } F^{-1}_{\overleftarrow{C}^{\pi_k}(x_{t-1},a_{t-1})}(\nu) + \\ &\left\langle \pi(\cdot|x_t), F^{-1}_{C^{\pi_k}(x_t,\cdot)}(1 - \nu + \tau) \right\rangle \le \frac{d}{g_c(\tau) H(g_c)}, \end{aligned} \tag{2}$$

where π_k is the current policy, $H(g_c) = \int_0^1 g_c(x) dx$, $\nu \in [0, 1]$ is the quantile level of realized cost $\sum_{t'=0}^{t'=t-1} c(x_{t'}, a_{t'})$, and $\forall \tau \in [0, \nu]$ is in the supports of g_c. Note that ν is contained in x_t as $x_t = (s_t, \nu)$ by definition. It could be sub-optimal in DCPI to substitute the original quantile function with the two decomposed parts given in Lemma 5.1, however, empirical results in various environments still show superior performance of our algorithm (Sec. 7). We aim to mitigate the optimality gap (if any) in our future works.

Moreover, we have the following guarantee on the feasibility of the solved policy π_{k+1} for problem (1). Formal guarantees with proofs are given in Appendix A.4.

Theorem 5.2 (Informal Guarantee)**.** *Assume each policy in Π_{nsta} is log-Lipschitz in τ with coefficient L. Assume $\frac{\pi_{k+1}(a|x)}{\pi_k(a|x)} \in [1 - \delta, 1 + \delta]$, $\pi_k(a|x) > 0, \forall x, a$ with $\delta \in [0, 1)$, and $L = L_\delta \delta$ for some $L_\delta \in \mathbb{R}_{>0}$. Then, if $g_c = Uniform([\xi, 1.0])$, $\mathbb{E}_{a_0 \sim \pi_{k+1}(\cdot|x_0)} \Phi_{g_c} \left[C^{\pi_{k+1}}(x_0, a_0) \right] \le \frac{d}{1-\delta} + D(\delta, \xi, T)$.*

In the theorem, $D(\delta, \xi, T)$ is the solution value to a constrained problem shown in Appendix A.4. The theorem suggests that, when L is linear to δ, and as δ tends to 0, $D(\delta, \xi, T)$ also tends to 0. Based on these findings, we have chosen to use PPO for smooth policy updates in our experiments, and we have selected very small values for the δ. Moreover, the theorem has no assumptions on the feasibility of π_k, which suggests that DCPI always projects the (possibly) infeasible policy to an approximately feasible one.

5.2 Practical Implementation

With the above theorem, we can safely solve the surrogate problem DCPI instead of solving the challenging problem (1) directly. Our

proposed algorithm for DCPI, Risk-Aware Constrained rl (RAC), is shown in Alg. 1. RAC alternates between the policy evaluation stage and the policy improvement stage, which are introduced in detail in the following sections.

Policy evaluation. In the policy evaluation stage, the action-value function Q_ς is trained by minimizing the squared TD-error [25]. The distributional cost critic $U_\eta(x, a, \tau)$ is instantiated with the IQN structure [12] to approximate the quantile function $F^{-1}_{C^{\pi_\theta}(x,a)}(\tau)$. The IBDC $I_\phi(x, a, \tau)$ is instantiated from class $\mathcal{F}_{linear}$ or $\mathcal{F}_{nn}$ to approximate the quantile function $F^{-1}_{\overleftarrow{C^{\pi_\theta}}(x,a)}(\tau)$.

Let $\mathcal{D} = \{(x_{i-1}, x_i, a_i, r_i, c_i, x_{i+1})\}_{i=1}^B$ be a random mini-batch with size B. Then, η is updated by minimizing the empirical Huber quantile regression loss $\frac{1}{B}\sum_{i=1}^B \mathcal{L}_\kappa(\beta_i, \tau_i)$, where

$$\mathcal{L}_\kappa(\beta_i, \tau_i) = \begin{cases} |\tau_i - \mathbf{1}(\beta_i < 0)| \cdot \beta_i^2/(2\kappa), & \text{if } |\beta_i| \le \kappa, \\ |\tau_i - \mathbf{1}(\beta_i < 0)| \cdot (|\beta_i| - \kappa/2), & \text{otherwise,} \end{cases} \tag{3}$$

and $\beta_i = c_i + U_{\eta'}(x_{i+1}, a_{i+1}, \tau_i') - U_\eta(x_i, a_i, \tau_i)$, $a_{i+1} \sim \pi_\theta(\cdot|x_{i+1})$. η' denotes the parameters of the target network. κ is Huber threshold, and $\tau_i, \tau_i' \sim \text{Uniform}([0, 1])$. The parameters ϕ of IBDC I_ϕ is updated in a similar way with $\beta_i = c_i + I_{\phi'}(x_{i-1}, a_{i-1}, \tau_i') - I_\phi(x_i, a_i, \tau_i)$, $a_{i-1} \sim \pi_\theta(\cdot|x_{i-1})$.

Policy improvement. In the policy improvement stage, RAC solves DCPI by introducing an additional Lagrangian multiplier $\lambda \ge 0$ to convert DCPI to an unconstrained min-max problem, and solves the latter one by alternatively updating λ and policy parameter θ until convergence. λ is updated by

$$\begin{aligned} \lambda' = \Gamma_{\ge 0}\Bigg[\lambda + \frac{\omega_\lambda}{B}\sum_{i=1}^B \big(I_\phi(x_i, a_i, \nu_i) - c_i + \\ U_\eta(x_i, a_i, 1 - \nu_i + \tau) - \frac{d}{g_c(\tau)H(g_c)}\big)\Bigg], \end{aligned} \tag{4}$$

where Γ is the projection operator, ω_λ is the learning rate, and $\tau \in [0, \nu_i]$ is sampled uniformly from supports of g_c. Note that we make a simplification here by instantiating $F^{-1}_{\overleftarrow{C^{\pi_k}}(x_{t-1},a_{t-1})}(\nu)$ in DCPI with $I_\phi(x_i, a_i, \nu_i) - c_i$ to make sampling decorrelated, where ν_i is the quantile level contained in $x_i = (s_i, \nu_i)$. Further, policy parameter θ is updated with the PPO algorithm [22], as it could restrict the ratio $\rho_i = \frac{\pi_\theta(a_i|x_i)}{\pi_{\theta_k}(a_i|x_i)}$ that is required in Thm. 5.2. The loss for policy network is $L_\theta = -\frac{1}{B}\sum_{i=1}^B \min\big(\rho_i A_i, \text{clip}(\rho_i, 1-\delta, 1+\delta)A_i\big)$, where $A_i = Q_\varsigma(x_i, a_i) - \lambda' \cdot \big(I_\phi(x_i, a_i, \nu_i) - c_i + U_\eta(x_i, a_i, 1 - \nu_i + \tau) - \frac{d}{g_c(\tau)H(g_c)}\big)$ is the shifted advantage function, and θ_k is the parameters of the current policy.

RAC Algorithm. Alg. 1 shows the RAC algorithm block. Firstly, the experience buffer, the Lagrangian multiplier and the parameters of (distributional and expected) critic networks and policy network are initialized (line 1). The execution of the policy is shown in line 3-12. At each time step t, action a_t is sampled according to $\pi_\theta(\cdot|x_t)$ (line 5). After executing a_t, the reward r_t, cost c_t and the next state s_{t+1} are obtained from the environment (line 6). Then, the accumulated cost C is updated (line 7). Line 8 infers the quantile level for the next state s_{t+1} by inputting C into $\overleftarrow{I}_\phi$, the inverse form of I_ϕ, as shown in Sec. 4.2. Line 9 concatenates the next state s_{t+1} and the inferred quantile level τ_{t+1} to constitute the input for the next round. Line 10 stores the whole transition into buffer. After replacing the old x_{t-1} with x_t, the iteration goes on until step T.

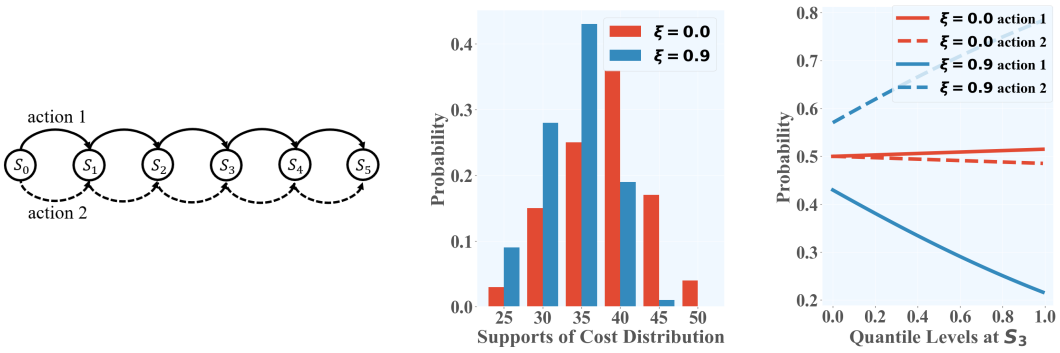
The updating stage of RAC is shown in line 13-18. A random batch is sampled firstly from the buffer (line 13). Then, it will be used to update the action-value function on reward using TD-error, and the distributional critic and IBDC on cost using Huber quantile regression losses (line 14). Then, the Lagrangian multiplier and the policy parameter are updated in sequence (line 15-16). Finally, the target networks are updated in a soft manner (line 17) and the buffer is emptied (line 18). RAC terminates when the policy converges.

Algorithm 1 Risk-Aware Constrained RL (RAC)

1: **Initialize:** buffer $\mathcal{B}$ as empty $\emptyset$; policy $\pi_\theta \in \Pi_{\text{nsta}}$; Lagrangian multiplier $\lambda = 0$; parameters of Q_ς; parameters of U_η; IBDC I_ϕ from class $\mathcal{F}_{linear}$ or $\mathcal{F}_{nn}$;
2: **repeat**
3: $\tau_0 = 1, C = 0, x_{t-1} =$ None, Concat $x_0 = (s_0, \tau_0)$;
4: **for** $t = 0$ **to** T **do**
5: Sample action $a_t \sim \pi_\theta(\cdot|x_t)$;
6: Execute a_t and obtain s_{t+1}, r_t, c_t;
7: Update accumulate cost $C = C + c_t$;
8: Infer quantile level of C: $\tau_{t+1} = \overleftarrow{I}_\phi(s_t, a_t, C)$;
9: Concat $x_{t+1} = (s_{t+1}, \tau_{t+1})$;
10: Store $(x_{t-1}, x_t, a_t, r_t, c_t, x_{t+1})$ into buffer $\mathcal{B}$;
11: Update $x_{t-1} = x_t$;
12: **end for**
13: Sample a random batch of B transitions $\mathcal{D} = \{(x_{i-1}, x_i, a_i, r_i, c_i, x_{i+1})\}_{i=1}^B$ from $\mathcal{B}$;
14: Update Q_ς by minimizing TD-error; update distributional cost critics U_η and IBDC I_ϕ with corresponding Huber quantile regression losses;
15: Update Lagrangian multiplier λ with equation (4);
16: Update policy parameter θ with loss L_θ;
17: Update target networks of critics;
18: Set buffer $\mathcal{B}$ as empty $\emptyset$;
19: **until** policy π_θ converges

6 RELATED WORKS

Recently, CMDPs, which search for policies to maximize expected reward return while satisfy constraints on the expected cost return, have raised a lot of attentions in constrained RL community [2–4, 21, 26, 33, 36]. However, the ignorance on the variance of the cost return makes even the optimal policies of CMDPs highly risky, especially in safety-critical applications. Several works have noticed such deficit, and they take the distribution of the long-term cost into consideration by formulating constraints under various risk measures [11, 32, 35, 39]. For example, Yang et al., Ying et al. utilize CVaR_α measure to constrain the expectation of the top α fraction of the long-term cost distribution below the threshold, while chance-constrained RL considered in [11] utilizes VaR_α measure to limit its probability below some threshold. Meanwhile, Luo and Ma, Sootla et al., Sootla et al., Thananjeyan et al., Wachi and Sui, Wang et al.



(a) (b) (c)

Figure 2: The TWS Environment is shown in Fig. 2(a). Fig.2(b) compares the cost distribution under different ξ; Fig.2(c) shows the trending of the probability of choosing action 1 and 2 at state S_3 when the input quantile level changes.

consider problems with hard constraints, whose goal is to maximize expected reward and limit the cost of any trajectory below the constraint threshold. In contrast, our RAC is not restricted to any specific risk measure, as discussed on the generality of RACO in Sec. 2. More importantly, RACO searches policies within the quantile-level-driven policy class Π_{nsta} instead of the common stationary policy class Π_{sta}, whose advantages are shown in Sec. 3 and Sec. 7 with various empirical studies.

Perhaps the most related works to ours are [8, 23, 24, 38, 39]. In [8, 38, 39], distributional critics are used to estimate the distribution of the cost return. While RAC also relies on distributional critics to estimate the future cost distribution, as described in detail in Sec. 5.2, the IBDC framework is completely novel as it estimates the backward cost distribution and is invertible to output the required quantile level. Moreover, Sootla et al., Sootla et al. also relies on the idea to augment the state space, where the left budget is recorded and added. However, as discussed in their paper, such augmentation method can only solve problems with hard constraints or expected constraints, while our RAC fills the gap and is able to solve RACOs with general risk measures g_c, as discussed in Sec. 2.

7 EXPERIMENTS

We conduct experiments in various environments to demonstrate the performance of RAC. We begin with a toy and a classic grid environment, which are friendly for visualizations, to present the advantages and rationality of the RAC algorithm and the IBDC framework. Later, we compare RAC with state-of-the-art algorithms in the safety gym benchmarks and two real-world applications.

7.1 Two-way Selection Environment

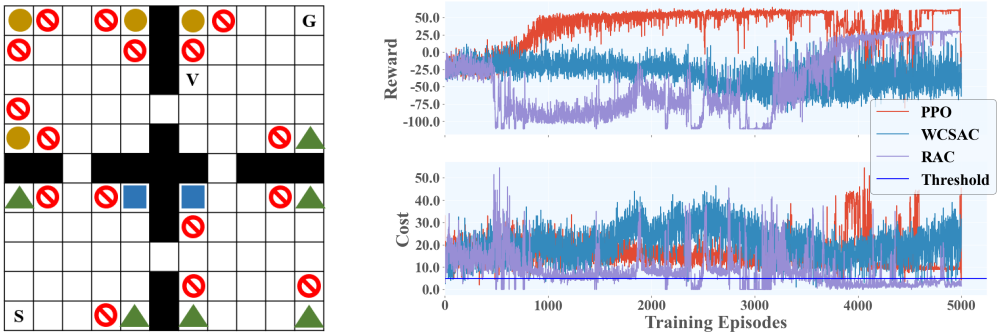
The two-way selection environment (TWS) is shown in Fig. 2(a). S_0 is the initial state and S_5 is the terminal state. At each non-terminal state, agent chooses action 1 or 2 and transits to the next state deterministically. Agent also obtains an immediate reward and cost 10 for taking action 1, and 5 for taking action 2. The threshold $d = 40$, and risk measure $g_c = \text{Uniform}([\xi, 1.0])$ with $\xi \in \{0.0, 0.9\}$. RAC utilizes linear function approximators for policy and class $\mathcal{F}_{linear}$ for IBDC. More settings and results are in Appendix B.2.

Fig. 2(b) shows the cost distribution of policies trained with RAC under different ξ. The figure shows that the cost distribution under $\xi = 0.0$ has much larger mass in range 40-50 comparing to the case when $\xi = 0.9$. Such difference makes sense as the RACO problem constrains the expectation of the top 10% in the cost distribution to be lower than 40 when $\xi = 0.9$, while it only constrains the expectation of the full cost distribution to be lower than 40 when $\xi = 0.0$. To get more intuitive interpretations on such difference, we visualize the probabilities to choose action 1 and 2 as the input quantile level changes at state S_3 in Fig. 2(c). When $\xi = 0.9$, the probability of selecting action 1 descends straightly as the quantile level increases, while it stays nearly unchanged when $\xi = 0.0$. In other words, when $\xi = 0.9$, the policy becomes much more conservative by choosing action 2 with higher probability when the accumulate cost gets higher before reaching state S_3. The distinct trending curves of different ξ suggest RAC is able to fit different risk levels and learn policies accordingly by manipulating the sensitivity on the input quantile level. Essentially, such visualization results validate our motivation for using non-stationary policies with quantiles as input, especially in the risk-averse setting.

7.2 Constrained Four Room Environment

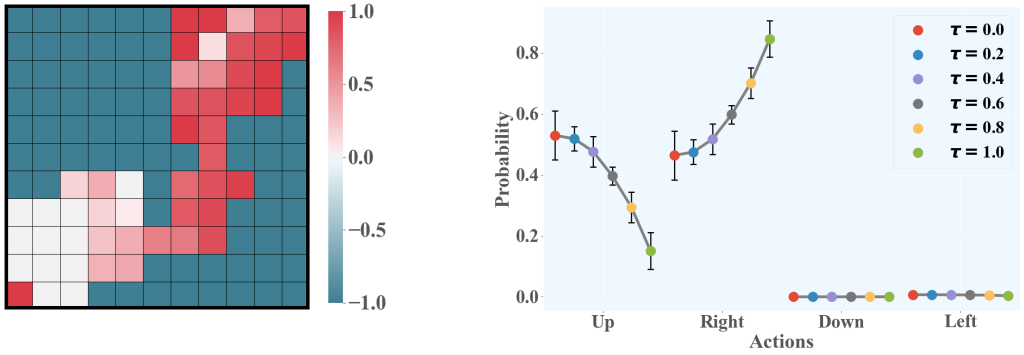
The constrained four room environment (CFR) is an extension of the classical four room environment [13], as shown in Fig. 3(a). Agent locates at grid S initially in CFR, and the goal grid is G. The total horizon of CFR is 100. At each step, agent can move to one of its neighboring grids with direction *up*, *right*, *down* or *left*. The move is successful only if the next grid is within CFR and is not a wall, i.e., black grid. Agent also gets reward -1 when unsuccessful move is made. When the next grid has the circle sign, the square sign or the triangle sign, agent gets reward 10, 5, 1, respectively. When the next grid has the stop sign, agent gets cost 1. The rewarding signs will disappear once being visited, while the stop signs will stay the same. The negative Manhattan distance from agent's location to grid G counts as the reward at step 100. The goal grid G is absorbing and generates reward 20 when agent is at grid G at step 100.

In CFR, the threshold $d = 5$ and risk measure $g_c = \text{Uniform}([0.5, 1.0])$. The unconstrained **PPO** algorithm and the state-of-the-art algorithm for solving CVaR-constrained problems, **WCSAC** [32], are compared here. RAC utilizes $\mathcal{F}_{nn}$ as the function class for IBDC. The training curves in Fig. 3(b) suggest that both PPO and WCSAC are constraint-violating as they have cost larger than 5 at the end of



(a) (b)

Figure 3: Fig.3(a) shows the CFR environment; Fig.3(b) shows the training curves of algorithms.



(a) (b)

Figure 4: Visualizations of the trained policy in 100 test episodes in CFR. Fig.4(a) shows quantile level heatmap along the trajectories, where unvisited grids have value -1; Fig.4(b) shows the action-selecting probabilities at grid V with different input quantile levels.

training, while RAC satisfies the constraint threshold and achieves the second-best in reward.

We also visualize the policy learned with the RAC algorithm to get more intuitions and interpretations. We test the learned policy with 100 episodes, and Fig. 4(a) shows the heatmap of quantile levels along the trajectories. Grids that are unvisited have value -1. Apparently, the quantile level is lower around the initial grid S, and gets larger when it moves closer to goal grid G, since the cost is also accumulated along the trajectory. Such quantile level is crucial for decision-making that is evidenced by Fig. 4(b), where we visualize the action-selecting probabilities at grid V when different quantile levels are inputted. As the quantile level increases from 0.0 to 1.0, the probability for selecting *up* decreases while the probability for selecting *right* increases. Such phenomenon is intuitive because when the quantile level is high, it means that the agent has already accumulated a large cost before reaching grid V. Hence, it becomes sub-optimal to still go up as it may violate the constraint threshold easily. Conversely, the agent has larger probability to go up than right when the quantile level is little as going up is more rewarding. Such distinct action-selecting probabilities when different quantile levels are inputted will adjust the policy to be conservative or adventurous, which in fact underlies its outperformance.

7.3 Safety Gym Benchmarks

We compare RAC with state-of-the-art algorithm for CVaR-constrained problems, **WCSAC** [32], in two safety gym benchmarks, PointGoal1 and CarGoal1. A cubic box is controlled in PointGoal1 and a tiny car is controlled in CarGoal1, and they aim to reach the destination point while avoiding unsafe regions. In both benchmarks, the threshold $d = 25$ and risk measure $g_c = \text{Uniform}([\xi, 1.0])$ with $\xi \in \{0.1, 0.5, 0.9\}$. $\mathcal{F}_{nn}$ is used for the IBDC class. More details of benchmarks and RAC are given in Appendix B.4. Table 1 shows the testing results of trained policies. For different ξ, we report the mean and std of long-term reward and the corresponding fraction of long-term cost in the table. The results suggest that RAC outperforms WCSAC as RAC satisfies all the constraints under different ξ in both benchmarks, while WCSAC violates the constraint heavily.

7.4 Simulations of Real-World Applications

Finally, we test RAC in two simulations of real-world applications with hard constraints: UAV Maneuvering (UAVM) [14, 31] and Budgeted Load Balancing (BLB). UAVM (Fig. 5) simulates the movement of UAV in an area with unsafe regions, which are represented as cylinders between the initial position and the destination. The goal of UAVM is to control UAV to reach the destination as possible while bypassing the unsafe regions.

BLB (Fig. 6) originates from the unconstrained load balancing environment [17], which simulates the load balancing task with heterogeneous servers in datacenters. The goal of BLB is to minimize the average job completion time, while limiting the maximal computational expense of servers below the budget. More details of environments can be found in Appendix B.5.

Table 1: Testing results in 100 episodes in safety gym benchmarks. W is short for WCSAC and R is short for RAC. The suffix of algorithms are values of ξ. The mean and std of reward, top 10% cost, top 50% cost and top 90% cost of the 100 testing episodes are given in the table. - represents the objective is not optimized under the algorithms.

Algs	PointGoal1				CarGoal1			
	Reward	10%	50%	90%	Reward	10%	50%	90%
W-0.1	$6.0_{\pm 4.2}$	-	-	$39.4_{\pm 14.0}$	$14.0_{\pm 9.1}$	-	-	$27.6_{\pm 69.6}$
R-0.1	$\mathbf{2.0_{\pm 2.1}}$	-	-	$\mathbf{15.9_{\pm 13.3}}$	$\mathbf{-5.2_{\pm 7.7}}$	-	-	$\mathbf{8.7_{\pm 10.7}}$
W-0.5	$2.3_{\pm 3.8}$	-	$32.0_{\pm 47.6}$	-	$7.0_{\pm 4.8}$	-	$42.8_{\pm 36.7}$	-
R-0.5	$\mathbf{-3.1_{\pm 3.7}}$	-	$\mathbf{16.3_{\pm 14.7}}$	-	$\mathbf{-22.5_{\pm 7.6}}$	-	$\mathbf{12.3_{\pm 7.0}}$	-
W-0.9	$\mathbf{4.1_{\pm 1.6}}$	$\mathbf{25.0_{\pm 20.6}}$	-	-	$5.1_{\pm 2.8}$	$46.2_{\pm 37.7}$	-	-
R-0.9	$-3.7_{\pm 2.3}$	$20.0_{\pm 3.6}$	-	-	$\mathbf{-28.8_{\pm 1.1}}$	$\mathbf{13.0_{\pm 0.8}}$	-	-

Table 2: Testing results (mean±std) in 100 episodes in UAVM and BLB.

Algorithms	UAV Maneuvering		Budgeted Load Balancing	
	Reward	Violation Rate	Reward	Violation Rate
PPO	-27.16±0.16	100%	-454.91±61.40	93.07%±0.25%
PPO-Lagrangian	-29.92±2.07	0%	-560.71 ± 87.11	46.19%±1.12%
GSBF	-29.10±2.55	0.4%±0.3%	-	-
Saute-n_1	-30.94±2.01	100%	-441.18±67.12	88.60%±1.61%
Saute-n_2	-33.26±0.25	100%	-453.68±66.50	91.94%±1.78%
Saute-n_3	-128.00±36.03	0%	-471.22±77.76	83.12%±1.77%
RAC	**-28.28±0.10**	**0%**	**-638.82±74.72**	**11.80%±2.02%**

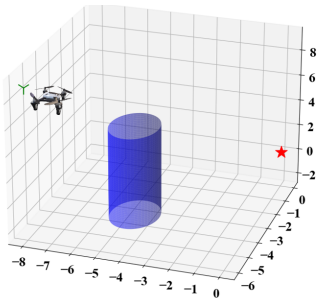


Figure 5: Illustration of the UAVM environment.

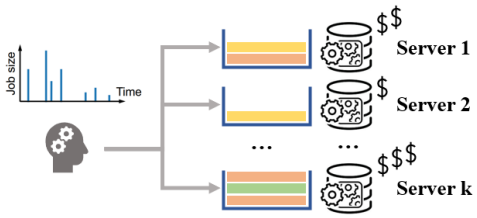


Figure 6: Illustration of the BLB environment.

In both UAVM and BLB, g_c is the Dirac delta distribution at 1. IBDC is initialized from class $\mathcal{F}_{nn}$ for RAC. The baselines for comparison are follows. **PPO** is an unconstrained algorithm to show the maximal reward when no constraints are imposed. **PPO-Lagrangian** is a classical algorithm for solving CMDP problems [1]. **GSBF** achieves state-of-the-art results in UAVM [31]. **Saute** [24] relies on state augmentations and is the state-of-the-art algorithm for solving hard constraint problems. In Saute, we also set three increasing hyper-parameters n_1, n_2, n_3 for each environment. Their detailed values are listed in Appendix B.5 as they are domain-specific.

The testing results of algorithms are summarized in Table 2. In UAVM, RAC, PPO-Lagrangian and RAC-n_3 have zero violations on safety, while RAC has the largest reward. In fact, RAC achieves the highest reward among all the constrained algorithms. In BLB, GSBF is not applicable here as the transition function and the unsafety regions of BLB are unknown. Among all applicable baselines, BLB achieves the minimal violation rate in budget constraint with slight conservativeness in reward. It turns out that Saute is sensitive to the hyper-parameter n from the results of two environments, and it needs domain specific knowledge to tune the optimal n.

8 CONCLUSION AND DISCUSSION

This paper targets at solving constrained RL problems, for which our major novelty lies in searching policies within the quantile-level-driven policy class and decomposing the challenging trajectory-level constraints into state-level constraints. As discussed in Sec. 2, our problem formulation RACO is general and applicable for various risk measures. Moreover, the experimental results in Sec. 7 demonstrate the effectiveness of RAC for real-world applications, such as maneuvering the UAV to avoid unsafe regions and controlling the expense in load-balancing tasks. Hence, we argue that our work has positive effects on the deployment of RL for wider real-world applications, especially those with safety-critical constraints.

ACKNOWLEDGMENTS

This work was supported by NSF China (No. U21A20519, U20A20181, 62372288, 62202298), and by DiDi GAIA Research Collaboration Plan.

REFERENCES

[1] Joshua Achiam and Dario Amodei. 2019. Benchmarking Safe Exploration in Deep Reinforcement Learning. https://api.semanticscholar.org/CorpusID:208283920. Accessed: 2023-08-12.

[2] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. 2017. Constrained policy optimization. In *ICML*.

[3] Eitan Altman. 1999. *Constrained Markov decision processes: stochastic modeling*. Routledge.

[4] Yarden As, Ilnura Usmanova, Sebastian Curi, and Andreas Krause. 2022. Constrained Policy Optimization via Bayesian World Models. In *ICLR*.

[5] Marc G Bellemare, Will Dabney, and Rémi Munos. 2017. A distributional perspective on reinforcement learning. In *ICML*.

[6] Felix Berkenkamp, Matteo Turchetta, Angela Schoellig, and Andreas Krause. 2017. Safe model-based reinforcement learning with stability guarantees. In *NIPS*.

[7] Craig Boutilier and Tyler Lu. 2016. Budget allocation using weakly coupled, constrained Markov decision processes. In *UAI*.

[8] Dan A Calian, Daniel J Mankowitz, Tom Zahavy, Zhongwen Xu, Junhyuk Oh, Nir Levine, and Timothy Mann. 2020. Balancing constraints and rewards with meta-gradient d4pg. In *ICLR*.

[9] Nicolas Carrara, Edouard Leurent, Romain Laroche, Tanguy Urvoy, Odalric-Ambrym Maillard, and Olivier Pietquin. 2019. Budgeted reinforcement learning in continuous state space. In *NIPS*.

[10] Yash Chandak, Scott Jordan, Georgios Theocharous, Martha White, and Philip S Thomas. 2020. Towards safe policy improvement for non-stationary MDPs. In *NIPS*.

[11] Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. 2017. Risk-constrained reinforcement learning with percentile risk criteria. *JMLR* (2017).

[12] Will Dabney, Georg Ostrovski, David Silver, and Rémi Munos. 2018. Implicit quantile networks for distributional reinforcement learning. In *ICML*.

[13] Michael Gimelfarb, André Barreto, Scott Sanner, and Chi-Guhn Lee. 2021. Risk-Aware Transfer in Reinforcement Learning using Successor Features. In *NIPS*.

[14] Wanxin Jin, Shaoshuai Mou, and George J Pappas. 2021. Safe pontryagin differentiable programming. In *NIPS*.

[15] Yuping Luo and Tengyu Ma. 2021. Learning barrier certificates: Towards safe reinforcement learning with zero training-time violations. In *NIPS*.

[16] Yecheng Ma, Dinesh Jayaraman, and Osbert Bastani. 2021. Conservative offline distributional reinforcement learning. In *NIPS*.

[17] Hongzi Mao, Shaileshh Bojja Venkatakrishnan, Malte Schwarzkopf, and Mohammad Alizadeh. 2019. Variance Reduction for Reinforcement Learning in Input-Driven Environments. *arXiv* (2019). arXiv:1807.02264 [cs.LG]

[18] Tetsuro Morimura, Eiji Uchibe, Junichiro Yoshimoto, Jan Peters, and Kenji Doya. 2009. Derivatives of Logarithmic Stationary Distributions for Policy Gradient Reinforcement Learning. *Neural computation* (2009).

[19] R.Tyrrell Rockafellar and Stanislav Uryasev. 2002. Conditional value-at-risk for general loss distributions. *Journal of Banking & Finance* (2002).

[20] R Tyrrell Rockafellar, Stanislav Uryasev, et al. 2000. Optimization of conditional value-at-risk. *Journal of Risk* (2000).

[21] Harsh Satija, Philip Amortila, and Joelle Pineau. 2020. Constrained Markov Decision Processes via Backward Value Functions. In *ICML*.

[22] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *arXiv* (2017). arXiv:1707.06347 [cs.LG]

[23] Aivar Sootla, Alexander Cowen-Rivers, Jun Wang, and Haitham Bou Ammar. 2022. Enhancing safe exploration using safety state augmentation. In *NIPS*.

[24] Aivar Sootla, Alexander I Cowen-Rivers, Taher Jafferjee, Ziyan Wang, David H Mguni, Jun Wang, and Haitham Ammar. 2022. Sauté rl: Almost surely safe reinforcement learning using state augmentation. In *ICML*.

[25] Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement Learning: An Introduction*. A Bradford Book.

[26] Chen Tessler, Daniel J Mankowitz, and Shie Mannor. 2019. Reward constrained policy optimization. In *ICLR*.

[27] Brijen Thananjeyan, Ashwin Balakrishna, Suraj Nair, Michael Luo, Krishnan Srinivasan, Minho Hwang, Joseph E Gonzalez, Julian Ibarz, Chelsea Finn, and Ken Goldberg. 2021. Recovery rl: Safe reinforcement learning with learned recovery zones. *RAL* (2021).

[28] Amos Tversky and Daniel Kahneman. 1992. Advances in prospect theory: Cumulative representation of uncertainty. *Journal of Risk and uncertainty* (1992).

[29] Akifumi Wachi and Yanan Sui. 2020. Safe reinforcement learning in constrained Markov decision processes. In *ICML*.

[30] Shaun S Wang. 2000. A class of distortion operators for pricing financial and insurance risks. *Journal of risk and insurance* (2000).

[31] Yixuan Wang, Simon Sinong Zhan, Ruochen Jiao, Zhilu Wang, Wanxin Jin, Zhuoran Yang, Zhaoran Wang, Chao Huang, and Qi Zhu. 2022. Enforcing Hard Constraints with Soft Barriers: Safe Reinforcement Learning in Unknown Stochastic Environments. *arXiv* (2022). arXiv:2209.15090 [eess.SY]

[32] Qisong Yang, T. D. Simão, Simon Tindemans, and Matthijs T. J. Spaan. 2021. WCSAC: Worst-Case Soft Actor Critic for Safety-Constrained Reinforcement Learning. In *AAAI*.

[33] Tsung-Yen Yang, Justinian Rosca, Karthik Narasimhan, and Peter J Ramadge. 2020. Projection-based constrained policy optimization. In *ICLR*.

[34] Zhaoxing Yang, Rong Ding, Haiming Jin, Yifei Wei, Haoyi You, Guiyun Fan, Xiaoying Gan, and Xinbing Wang. 2021. DeCOM: Decomposed Policy for Constrained Cooperative Multi-Agent Reinforcement Learning. *arXiv* (2021). arXiv:2111.05670 [cs.LG]

[35] Chengyang Ying, Xinning Zhou, Hang Su, Dong Yan, Ning Chen, and Jun Zhu. 2022. Towards Safe Reinforcement Learning via Constraining Conditional Value-at-Risk. *arXiv* (2022). arXiv:2206.04436 [cs.LG]

[36] Haonan Yu, Wei Xu, and Haichao Zhang. 2022. Towards Safe Reinforcement Learning with a Safety Editor Policy. In *NIPS*.

[37] Ming Yu, Zhuoran Yang, Mladen Kolar, and Zhaoran Wang. 2019. Convergent policy optimization for safe reinforcement learning. In *NIPS*.

[38] Hengrui Zhang, Youfang Lin, Sheng Han, Shuo Wang, and Kai Lv. 2022. Conservative Distributional Reinforcement Learning with Safety Constraints. *arXiv* (2022). arXiv:2201.07286 [cs.LG]

[39] Jianyi Zhang and Paul Weng. 2021. Safe distributional reinforcement learning. In *DAI*.