RETHINKING PARETO APPROACHES IN CONSTRAINED REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Constrained Reinforcement Learning (CRL) burgeons broad interest in recent years, which pursues both goals of maximizing long-term returns and constraining costs. Although CRL can be cast as a multi-objective optimization problem, it is still largely unsolved using standard Pareto optimization approaches. The key challenge is that gradient-based Pareto optimization agents tend to stick to known Pareto-optimal solutions even when they yield poor returns (i.e., the safest self-driving car that never moves) or violates the constraints (i.e., the record breaking racer that crashes the car). In this paper, we propose a novel Pareto optimization method for CRL with two gradient recalibration techniques to overcome the challenge. First, to explore around feasible Pareto optimal solutions, we use gradient re-balancing to let the agent improve more on under-optimized objectives at each policy update. Second, to escape from infeasible solutions, we propose gradient perturbation to temporarily sacrifice return to save costs. Experiments on the SafetyGym benchmarks show that our method consistently outperforms previous CRL methods in return while satisfying the cost constraints.

1 INTRODUCTION

By virtue of the close relationship to real-world applications, Constrained Reinforcement Learning (CRL) burgeons broad interest in recent years. Unlike the traditional RL, which targets maximizing cumulative rewards only, CRL pursues rewards while satisfying specific constraints (Achiam et al., 2017; Ding et al., 2020; Wachi & Sui, 2020; Satija et al., 2020). For example, in the scenario of auto-pilot, the well-trained agent should arrive at the destination accurately and meets the safety constraints in the meantime (Kong et al., 2021).

Most existing works formulate the CRL problem as a Constrained Markov Decision Process (CMDP) (Altman, 1999), which incorporates constraints and rewards into the same framework. Aside from returning a scalar reward after each action like conventional MDPs, CMDPs send back one or multiple cost signals independent of reward. Constraints are expressed explicitly in CMDPs by limiting the expected sum of each cost in the corresponding region.

Essentially, the purpose of CRL is to maximize rewards while controlling costs, which would be naturally associated with the Multi-objective optimization (Deb, 2014). In recent years, Pareto approaches (Sener & Koltun, 2018; Lin et al., 2019), which find a steep gradient that benefits to all objective, have been generally leveraged to multi-objective optimization. The ultimate goal of Pareto approaches is finding a Pareto-optimal (Pareto, 1897) solution, in which no objective can be advanced without harming any other objectives. However, algorithms for CRL seldom consider experience from the Pareto optimization area because existing Pareto approaches perform poorly in practical CRL problems (Tessler et al., 2018).

According to our analysis, existing Pareto approaches are not practical at CRL because they can only find trivial Pareto-optimal policy. As shown in Fig.1(b), when the gradients to optimize reward and cost disagree, Pareto approaches will synthesize a new gradient biased to the shorter one in direction. Suppose the policy is near-optimal about one objective (i.e. short gradient) while underdeveloped in another objective. In that case, this biased gradient will instead pay more attention to the objective with better performance. Policy updated by this biased gradient will be either too risky to be aware of the constraint on cost or too conservative to interact with the environment, which leads to an imbalanced development in rewards and costs. Meanwhile, restricted by the feature of simultaneous

improving all objectives, Pareto approaches cannot sacrifice one objective in exchange to advance another, which is a necessary skill to find a policy feasible to cost constraint in CRL. Thus, though existing Pareto approaches are able to find Pareto-optimal policy, the results not only have imbalance performance in terms of rewards and costs, but also are unable to guarantee that the constraint is met.

To tackle the aforementioned defects, we import gradient re-balancing and gradient perturbation mechanism to apply Pareto approaches in CRL. Gradient re-balancing re-defines the length of gradient and ensures we focus more on the objective in need. While gradient perturbation forces the Pareto optimizer to take confined attention on reward to improve cost by a bounded extent.

In this paper, we propose a novel CRL paradigm from the perspective of Pareto-optimal. With definitions on CRL and Pareto-optimal (Section 2), we rethink the connections between existing CRL methods and the concept of Pareto-optimal (Section 3.1). Then we analyze the pros and cons of applying Pareto optimization approaches to the CRL problem (Section3.2). Furthermore, we design a practical algorithm for CRL (Section 4) named CONTROL (abbr. for Constraints adaptive Pareto Reinforcement Learning) with two radient recalibration techniques. At last, we conduct experiments on a benchmark of CRL, the SafetyGym environment, the results of which demonstrate the superiority of CONTROL comparing to the state-of-the-art baselines (Section 5).

2 PRELIMINARY

2.1 CONSTRAINED MARKOV DECISION PROCESS

Markov Decision Process A normal Markov Decision Process (Sutton & Barto, 1998) can be described as a quadruple (S, A, P, R). Precisely, S denotes the state set; A denotes the action set; P is the distribution returning the probability of transiting to s' assuming we take action a in s, denoted as $P(s'|s, a)$; $R : S \times A \times S \to \mathbb{R}$ is the reward function, which delivers reward r as soon as the transition $s \to s'$ is accomplished. We make decisions for choosing actions by a policy $\pi : S \to \Delta_A$, which is a distribution over A. In this work, we parameterize our policy π_ω by a neural network with parameters $\omega \in \mathbb{R}^k$.

In an MDP, we take an action $a \sim \pi$ from initial state $s_0 \sim \rho_0(s_0)$ iteratively, and transit to a new state according to P, yielding a finite or infinite trajectory $\tau = (s_0, a_0, r_1, s_1, a_1, r_2, \ldots) \sim \pi$. Given a policy π, we are able to evaluate the goodness of a state or action by state-value function $V(s)$, action-value function $Q(s, a)$, and advantage-value function $A(s, a)$:

$$V_R^\pi(s_t) = \mathbb{E}_{a_t, s_{t+1}, \ldots}\left[\sum_{l=0}^{\infty} \gamma^l r_{t+l}\right], \quad Q_R^\pi(s_t, a_t) = \mathbb{E}_{s_{t+1}, a_{t+1}, \ldots}\left[\sum_{l=0}^{\infty} \gamma^l r_{t+l}\right], \tag{1}$$
$$A_R^\pi(s, a) = Q_R^\pi(s, a) - V_R^\pi(s).$$

In Eq(1), $\gamma \in [0, 1]$ is the discount factor, which weighs the future reward and instant reward. The goal of reinforcement learning is to discover an optimal policy π^* for MDP by solving:

$$\arg\max_\pi \mathbb{E}_{s_0 \sim \rho_0, \tau \sim \pi}\left[V_R^\pi(s_0)\right]. \tag{2}$$

Constrained MDP In this paper, we concentrate on CMDP with only one kind of cost, which is consistent with the settings in Achiam et al. (2017); Tessler et al. (2018); Yang et al. (2019). The main difference between CMDP and MDP is CMDP has a cost function $C : S \times A \times S \to \mathbb{R}$. Therefore, the feedback of the environment in one transition is a vector $(r, c) \in \mathbb{R}^2$, where $c \in \mathbb{R}_+$ is the value of cost. Similarly, we have value functions for cost: $V_C^\pi(s_t), Q_C^\pi(s_t, a_t), A_C^\pi(s, a)$ by switching the reward r to the cost c in Eq(1).

Formally, the policy optimization problem in CMDP is

$$\begin{aligned} &\max_\pi \; J_R(\pi) = \mathbb{E}_{s_0 \sim \rho_0, \tau \sim \pi}\left[V_R^\pi(s_0)\right], \\ &\text{s.t.} \quad J_C(\pi) = -\mathbb{E}_{s_0 \sim \rho_0, \tau \sim \pi}\left[V_C^\pi(s_0)\right] \geq \zeta, \end{aligned} \tag{3}$$

where $\zeta < 0$ denotes the predefined constraint threshold. In Eq.(3), $J_R(\pi)$ and $J_C(\pi)$ represents the performance of π with respect to reward and cost, respectively. Specifically, here we let $J_C(\pi)$ negative for readability.

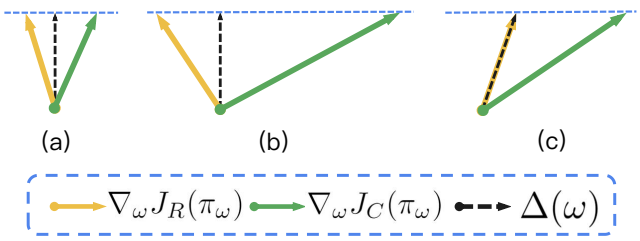


Figure 1: An illustration of Pareto direction $\Delta(\omega)$.
(a): $\Delta(\omega)$ generalizes both gradients when they converge in direction.
(b): $\Delta(\omega)$ is biased to the shorter gradient when $\nabla_\omega J_R(\pi_\omega)$ and $\nabla_\omega J_C(\pi_\omega)$ disagree in direction.
(c): $\Delta(\omega)$ coincides with one gradient in some cases.

2.2 PARETO-OPTIMAL

To make clearer definitions, we introduce related concepts under the problem settings of CRL directly. To understand Pareto-optimal, we need a rule to compare which policy is better first.

Definition 2.1 (Dominate)**.** *For two policies π, π', we say π **dominates** π', denoted as $\pi \succ \pi'$ i.i.f $J_R(\pi) \geq J_R(\pi'), J_C(\pi) \geq J_C(\pi')$ and and at least one inequation is strictly holds.*

By Definition 2.1, we know that a policy π is better than π' when π is not worse than π' over reward and cost, and outperforms π' on at least one objective.

Definition 2.2 (Pareto-optimal Policy)**.** *We call an policy π_ω is **global Pareto-optimal** i.i.f. $\forall \omega' \in \mathbb{R}^k, \pi_{\omega'} \succ \pi_\omega$ is invalid, i.e. $\forall \omega' \in \mathbb{R}^k, \pi_{\omega'} \not\succ \pi_\omega$;We call an policy π_ω is **local Pareto-optimal** i.i.f. there exists a neighborhood $U \subset \mathbb{R}^k$ of ω s.t. $\forall \omega' \in U, \pi_{\omega'} \succ \pi_\omega$ is invalid, i.e. $\forall \omega' \in U, \pi_{\omega'} \not\succ \pi_\omega$.*

Without otherwise specification, we use Pareto-optimal referring to local Pareto optimal hereafter. A local Pareto-optimal policy is guaranteed to be a global Pareto-optimal policy only if $J_R(\pi_\omega)$ and $J_C(\pi_\omega)$ are concave over ω in $\mathbb{R}^k$, which is invalid in most Deep RL setting. From the angle of optimizing target, we notice that if π is a Pareto-optimal policy we cannot advance any $J(\pi)$ while keeping the performance of another. This leads to the definition of Pareto direction:

Definition 2.3 (Pareto direction)**.** *Given a parameterized policy π_ω, if an vector $\mathbf{v} \in \mathbb{R}^k$ is an updating direction of ω which can boost at least one $J(\pi_\omega)$ without harming another $J(\pi_\omega)$, then $\mathbf{v}$ is a Pareto direction of π_ω. Namely, if $\langle \nabla_\omega J_R(\pi_\omega), \mathbf{v} \rangle, \langle \nabla_\omega J_C(\pi_\omega), \mathbf{v} \rangle \geq 0$, then $\mathbf{v}$ is a **Pareto direction** of π, where $\langle \, , \, \rangle$ is the standard inner product and at least one inequation strictly holds.*

Pareto-optimal Searching To search a Pareto-optimal policy, the most straightforward idea is designing an iteration $\omega' = \omega + \eta(\omega)\Delta(\omega)$, where $\Delta(\omega)$ is a Pareto direction of ω, and $\eta(\omega) \in \mathbb{R}_+$ is stepsize. With appropriate η, we could ensure $\pi_{\omega'} \succ \pi_\omega$ in each updating until π_ω is Pareto-optimal. Intuitively, $\Delta(\omega)$ should be a linear combination of gradients of $J_R(\pi_\omega), J_R(\pi_\omega)$, i.e. $\Delta(\omega) = \beta_R \nabla_\omega J_R(\pi_\omega) + \beta_C \nabla_\omega J_C(\pi_\omega)$, in which β_R, β_C are called **Pareto weights**.

Both Fliege & Svaiter (2000) and Désidéri (2012) are established works for Pareto-optimal searching. Under the problem settings of CRL, the Pareto weights are obtained in Désidéri (2012) by solving following Quadratic Programming (QP):

$$\begin{aligned} \min_{\beta_R, \beta_C \in \mathbb{R}} \; & ||\beta_R \nabla_\omega J_R(\pi_\omega) + \beta_C \nabla_\omega J_C(\pi_\omega)||_2^2 \\ s.t. \;\; & \beta_R, \beta_C \geq 0, \;\; \beta_R + \beta_C = 1. \end{aligned} \tag{4}$$

Fig.1 illustrates the geometric relationship among $\nabla_\omega J_R(\pi_\omega)$,$\nabla_\omega J_C(\pi_\omega)$ and $\Delta(\omega)$ by three cases. As a result of constraints in Problem(4), $\Delta(\omega)$ ends on the line segment determined by endpoints of $\nabla_\omega J_R(\pi_\omega)$ and $\nabla_\omega J_C(\pi_\omega)$. Since we minimize the length of $\Delta(\omega)$, $\Delta(\omega)$ is normally a perpendicular vector of $\nabla_\omega J_R(\pi_\omega) - \nabla_\omega J_C(\pi_\omega)$ (Fig.1a,b) and sometimes it coincides with one of the gradient vectors (Fig.1c).

In Appendix A, we first re-elaborate the algorithms proposed in Fliege & Svaiter (2000) and Désidéri (2012) under the CRL framework. Furthermore, we provide concise proofs for their effectiveness in finding Pareto direction and show that they are fundamentally identical in CRL problems. Finally, we make completeness proof for possible extreme situations when applying Pareto-optimal in CRL.

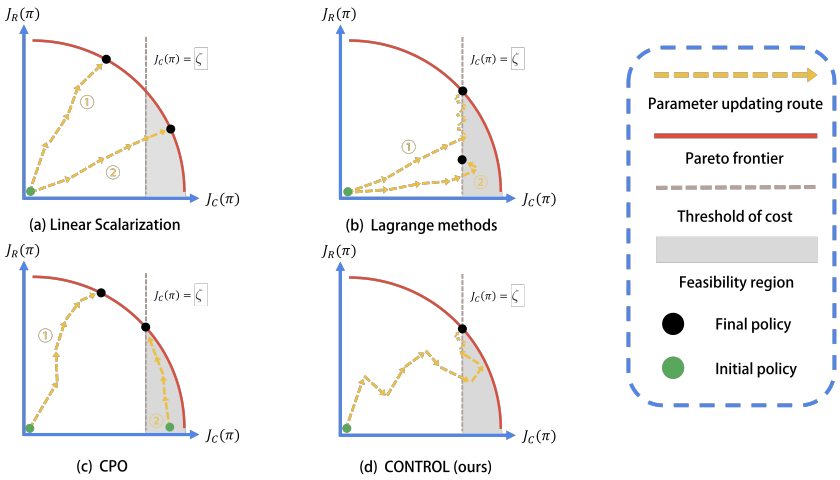


Figure 2: An illustration of how various methods find Pareto-optimal policies. **(a)**: Linear scalarization can find Pareto-optimal policy but need may not be Pareto-feasible; **(b)**: Lagrangian methods can find feasible policy, while they may fails for too conservative to explore the environment; **(c)**:CPO can find Pareto-optimal policy but cannot find Pareto-feasible policy if initial policy is too fat to reach the feasibility region; (d) Our method can find Pareto-feasible policy, while not each step is a Pareto direction.

3 Pareto-optimal in CRL

3.1 Connections to Prior CRL Works

The mainstream approaches to solve such problems could be grouped into two genres: (i) Lagrangian methods (Borkar, 2005; Tessler et al., 2018; Stooke et al., 2020); (ii) Trust Region methods (Achiam et al., 2017; Yang et al., 2019). Lagrangian methods combine primal reward-oriented and cost-oriented objectives into one dual min-max problem, converting CMDPs into unconstrained MDPs. While Trust Region methods attempt to determine a trust region in parameter space, where policies updated inside would not violate constraints with worst-case bound on reward performance.

Prior CRL algorithms are fundamentally finding Pareto-optimal policy (See proofs in Appendix C), which is a conclusion that unifies our work and existing works in methodology. To be consequentialism, any non-Pareto-optimal policy is unworthy of being regarded as a solution to a CMDP, since there must be a better one. Nevertheless, not every Pareto-optimal policy makes sense in a CMDP. What we truly need is Pareto-feasible policy:

Definition 3.1 (Pareto-feasible Policy)**.** *We call an policy π_ω is **Pareto-feasible** if π_ω is local Pareto-optimal while satisfying constraints.*

Fig.2 (a), (b) and (c) illustrate how prior CRL algorithms search Pareto-optimal policy. However, only in some occasions they are capable of finding a Pareto-feasible policy. In Fig.2 (a), we scalarize (r, c) per transition with random or preset weights linearly. With diverse selection of weights, the final policy may be different. Under such conditions, the search route could only reach a Pareto-feasible policy with weights good enough (route 2). As shown in Fig.2 (b), Lagrangian methods ensures that $J_C(\pi)$ is stable near the threshold. But they may fail to search Pareto-optimal policy because they are too conservative to explore and fall into local optima (route 2). For CPO (Achiam et al., 2017) in Fig.2 (c), it is able to maintain the policy within the feasibility region while increase $J_R(\pi)$ (route 1). Yet, it may fail to satisfy constraint when initial policy is not feasible. Results in experiments and proofs in Appendix C could corroborate that above interpretation is not heuristic.

3.2 Pros and Cons for Pareto-optimal in CRL

Advantages of Pareto-optimal Searching in CRL Based on the preceding analysis, we conclude three advantages for Pareto-optimal searching in CRL:

- **Preeminent**: Any Pareto-optimal policy ensures its superiority considering both reward and cost.
- **Reachable**: Pareto-optimal policy is not unique, and even simple algorithms (e.g. Linear Scalarization) can reach it.
- **Knowledge-free**: No prior knowledge is needed to apply Pareto approaches in CRL.

Disadvantages of Pareto-optimal Searching in CRL Despite noting the advantages of Pareto-optimal Searching, we found two main disadvantages from practical.

First, existing Pareto-optimal algorithms optimize all objectives with consistent extent (Proven in Theorem 4.2), which will lead to an **imbalanced development** of $J_R(\pi_\omega)$ and $J_C(\pi_\omega)$. This fact implies that the updating in this iteration would focus on rewards, which is already the better-optimized objective comparing to cost. Suppose our policy π_ω is in a situation where $J_R(\pi_\omega)$ is near the local optima while $J_C(\pi_\omega)$ is still under-optimized. In this circumstance, $\nabla_\omega J_C(\pi_\omega)$ is steeper and longer than $\nabla_\omega J_R(\pi_\omega)$, which is similar to Fig.1(b). As demonstrated in Fig.1(b), the Pareto direction $\Delta(\omega)$ is biased to $\nabla_\omega J_R(\pi_\omega)$ and its component in the direction of $\nabla_\omega J_R(\pi_\omega)$ is longer than $\nabla_\omega J_C(\pi_\omega)$'s. In extreme cases, if our policy reaches or approaches a trivial Pareto-optimal policy, it has no chance to escape and find feasible policy.

Second, existing Pareto-optimal approaches **overemphasize the simultaneous growth** of two objectives. In fact, in CRL, we need to sacrifice $J_R(\pi_\omega)$ in exchange for improvement in $J_C(\pi_\omega)$ to satisfy constraint in certain situations. Suppose we already reached Pareto frontier with a non-feasible policy. Instead of finding a Pareto direction to advance reward and cost, we prefer to search policies with less reward performance. Similarly, when our policy is too conservative, we should encourage it to take risks and pursue higher rewards. Thus, this defect will restrict our control to the parameter updating route and finally return a trivial policy without guarantee of satisfying constraint.

4 CONTROL: CONSTRAINTS ADAPTIVE PARETO RL

4.1 GRADIENT RECALIBRATION MECHANISMS

Gradient Re-balancing This is the corresponding improvement for imbalanced development issue. To tackle this issue, we should adapt $\nabla_\omega J_R(\pi_\omega)$ and $\nabla_\omega J_C(\pi_\omega)$ to achieve a similar degree of improvement in reward and cost at each iteration of policy parameters. Motivated by recent works about normalization in gradient (Chen et al., 2018; Mahapatra & Rajan, 2020), we reform Problem(4) as:

$$\begin{aligned} \min_{\beta_R,\beta_C\in\mathbb{R}} \; & ||\beta_R \nabla_\omega^N J_R(\pi_\omega) + \beta_C \nabla_\omega^N J_C(\pi_\omega)||_2^2, \\ s.t. \;\; & \beta_R, \beta_C \geq 0, \;\; \beta_R + \beta_C = 1, \end{aligned} \tag{5}$$

where $\nabla_\omega^N J_R(\pi_\omega), \nabla_\omega^N J_C(\pi_\omega)$ are normalized gradients and defined as:

$$\nabla_\omega^N J_R(\pi_\omega) = \frac{\nabla_\omega J_R(\pi_\omega)}{||\nabla_\omega J_R(\pi_\omega)||_2^2}, \nabla_\omega^N J_C(\pi_\omega) = \frac{\nabla_\omega J_C(\pi_\omega)}{||\nabla_\omega J_C(\pi_\omega)||_2^2}. \tag{6}$$

In Eq (5), the length of normalized gradient vectors becomes the reciprocal of its original length, which makes the original longer vector shorter. With solution $\beta_N^* = (\beta_R^N, \beta_C^N) \in \mathbb{R}_+^2$, we can find a better Pareto direction comparing to the Pareto direction derived from Problem (4).

Lemma 4.1. *If π_ω is not Pareto-optimal, $\Delta_N(\omega) := \beta_R^N \nabla_\omega J_R(\pi_\omega) + \beta_C^N \nabla_\omega J_C(\pi_\omega)$ is a Pareto direction of π_ω.*

Theorem 4.2. *Suppose the iteration paradigm is $\omega' = \omega + \eta(\omega)\Delta_N(\omega)$ and $\eta(\omega) \to 0$. Then:* ***(i)*** *if we use $\Delta(\omega)$ derived from Problem (4), the improvements in reward and cost are consistent. Specifically, when $\beta_R^* \in (0,1)$, $\dfrac{J_R(\pi_{\omega'}) - J_R(\pi_\omega)}{J_C(\pi_{\omega'}) - J_C(\pi_\omega)} \to 1$.* ***(ii)*** *if we use $\Delta_N(\omega)$ derived from Problem (5), the improvements in reward and cost are proportional to the square length of corresponding gradient. Specifically, when $\beta_R^N \in (0,1)$, $\dfrac{J_R(\pi_{\omega'}) - J_R(\pi_\omega)}{J_C(\pi_{\omega'}) - J_C(\pi_\omega)} \to \dfrac{||\nabla_\omega J_R(\pi_\omega)||_2^2}{||\nabla_\omega J_C(\pi_\omega)||_2^2}$.*

The proofs to Lemma 4.1 and Theorem 4.2 are provided in Appendix D.

According to Theorem 4.2, now we can pay more attention to the objective which needs more optimization. Note that it is imbalanced even if the improvement ratio is 1, because in practical the scales of $J_R(\pi), J_C(\pi)$ may be quite different.

Gradient Perturbation This is the corresponding improvement for assisting us to sacrifice rewards to satisfy constraint. A naive method for this idea is only optimizing cost when constraint is violated. But this is not realistic because: (i) focusing on cost too much may drive the agent to be too conservative to explore and fall into local optima; (ii)sometimes reward and cost are not competitive, totally ignoring reward is unwise.

In fact, manipulating Pareto weights can achieve the goal of sacrificing rewards when necessary. If current Pareto direction is unable to promote cost to a desirable extent, we must pay more attention to cost by raising β_C. Motivated by PPO (Schulman et al., 2017), we design a mechanism to control Pareto weight: when β_R is too big, we clip it to a smaller number. Since $\beta_R + \beta_C = 1$, the range of β_R can influence β_C, and an upper bound for β_R is also a lower bound for β_C.

Specifically, given a clipping threshold $t \in [0, 1]$, the original Pareto weight β_R^N will be clipped to $min(t, \beta_R^N)$. With fixed $\eta(\omega) \to 0$, we can deduce a lower bound for the growth of $J_C(\pi)$:

Theorem 4.3 (The lower bound of $J_C(\pi)$ improvement)**.** *Given the parameter updating paradigm $\omega' = \omega + \eta(\omega)\Delta_N(\omega)$ and clipping threshold a, we have the lower bound for $J_C(\pi'_\omega) - J_C(\pi_\omega)$:*

$$\begin{aligned} &J_C(\pi_{\omega'}) - J_C(\pi_\omega) \geq \\ &\eta(\omega)\left[t||\nabla_\omega J_C(\pi_\omega)||_2^2\langle\nabla_\omega^N J_R(\pi_\omega) - \nabla_\omega^N J_C(\pi_\omega), \nabla_\omega^N J_C(\pi_\omega)\rangle + 1\right] - CD_{\mathrm{KL}}^{\max}\left(\pi_{\omega'}, \pi_\omega\right), \end{aligned} \tag{7}$$

where $C = 4\epsilon\gamma/(1-\gamma)^2$ and $\epsilon = max_s|\mathbb{E}_{a\sim\pi_{\omega'}}\left[A^{\pi_\omega}(s,a)\right]|$. And this bound is strictly positive related to t.

The proof to Theorem 4.3 are provided in Appendix D, in which we also prove that this lower bound is tighter than the situation without clipping. Specifically, this theorem holds if swapping J_C, J_R.

The clipping operation ensures enough Pareto weight on the objective underdeveloped and perturb the weight if not meeting the clipping threshold. As clipping operation is imported to Problem (5), $\Delta_N(\omega)$ may not be a Pareto direction. But it ensures a lower bound for the improvement on $J_C(\pi)$, which is vital to find a feasible solution. To obtain a nice lower bound, we must choose proper t and $\eta(\omega)$. For t we can search by grid and search $\eta(w)$ by line backtracking (Armijo, 1966).

4.2 Practical Implementation

Base RL model Our method is adaptable and can be applied with any policy-gradient-based RL algorithm. In this paper, we adopt Actor-critic-based PPO (Schulman et al., 2015) as the base model of CONTROL. To estimate value functions for both reward and cost, we have two critics to approach $Q_R^\pi\left(s_t, a_t\right), Q_C^\pi\left(s_t, a_t\right)$, separately. Under the framework of PPO, $\nabla_\omega J_R(\pi_\omega), \nabla_\omega J_C(\pi_\omega)$ is determined as:

$$\nabla_\omega J_R(\pi_\omega) = \frac{\partial \mathbb{E}_{s\sim\pi_{old}, a\sim\pi_\omega}\left[A_R^{\pi_{\mathrm{old}}}(s,a)\right]}{\partial\omega}, \nabla_\omega J_C(\pi_\omega) = \frac{\partial \mathbb{E}_{s\sim\pi_{old}, a\sim\pi_\omega}\left[-A_C^{\pi_{\mathrm{old}}}(s,a)\right]}{\partial\omega}. \tag{8}$$

Pareto-Optimal to Pareto Feasible With Theorem 4.3, policy updating in CONTROL is guaranteed to converge to a Pareto-optimal optimal policy with a fixed clipping threshold t. However, this policy may not be Pareto-feasible. In this case, a search for t and $\eta(\omega)$ is necessary but this will affect the efficiency of our algorithm.

To choose t and $\eta(\omega)$ with efficiency, we devise a heuristic mechanism which keeps $\eta(\omega)$ as a constant and changes t according to the change of $J_R(\pi), J_C(\pi)$. If the improvement of the performance about cost in one iteration is not good enough, we decrease t by a little quantity to make a stricter clip. Similarly, if the performance about reward has not been improved within certain epochs, we increase t by identical quantity. Particularly, we clip β_C^N to encourage risky exploration when the performance about cost reflects that our policy is too conservative. The indicators of performance are acquired in an on-policy way.

We introduce how CONTROL trains an agent and modifies t in a pseudo-code, which can be found in Appendix E.

Model	Goal-Lvl 1		Goal-Lvl 2		Button-Lvl 1		Button-Lvl 2		Push-Lvl 1		Push-Lvl 2	
	Reward	Cost	Reward	Cost	Reward	Cost	Reward	Cost	Reward	Cost	Reward	Cost
TRPO	25.12	56.14	23.58	213.17	25.83	138.36	25.59	172.61	7.90	47.11	4.69	75.16
PPO	24.99	58.09	22.27	198.65	26.01	150.12	23.93	188.17	3.41	67.12	2.21	71.25
TRPO_L	17.03	25.47	5.49	25.27	8.18	32.59	3.73	22.51	4.19	26.31	1.30	23.38
PPO_L	13.58	14.21	1.15	31.66	4.84	23.01	2.38	17.99	2.15	40.20	1.52	17.27
CPO	23.21	42.52	14.37	60.11	18.25	80.25	16.78	74.43	7.21	38.94	1.84	29.37
MGDA	25.61	40.15	7.31	60.91	6.02	54.62	1.99	11.20	0.80	16.93	0.89	29.77
CONTROL	**21.86**	23.12	**8.39**	20.84	**9.38**	22.67	**6.44**	23.83	**2.39**	17.17	**2.98**	20.92
CONTROL-R	13.21	30.80	1.51	50.57	5.94	35.29	2.18	17.21	0.75	8.89	1.61	17.75
CONTROL-P	17.89	45.22	2.23	42.49	3.90	52.05	3.50	54.74	1.36	31.11	2.25	28.77

Table 1: Comparison Results of CONTROL and other baselines with cumulative threshold <25.

Model	Goal-Lvl 1		Goal-Lvl 2		Button-Lvl 1		Button-Lvl 2		Push-Lvl 1		Push-Lvl 2	
	Reward	Cost	Reward	Cost	Reward	Cost	Reward	Cost	Reward	Cost	Reward	Cost
TRPO_L	24.57	28.01	8.72	50.15	12.90	48.97	7.27	69.21	**6.13**	44.72	2.09	54.41
PPO_L	25.11	26.12	4.73	63.02	10.65	73.15	4.14	62.02	3.67	33.62	1.62	41.74
CONTROL	**25.29**	40.68	**16.38**	48.41	**14.66**	46.57	**13.89**	48.81	4.15	44.71	**2.52**	43.03

Table 2: Comparison Results of CONTROL and other baselines with cumulative threshold <50.

5 EXPERIMENTS

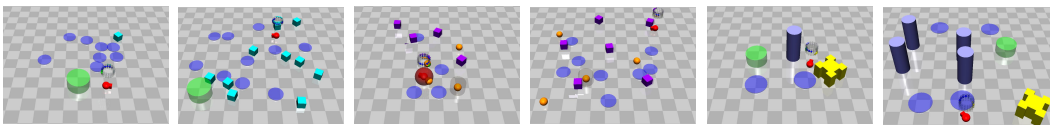
We test CONTROL in SafetyGym (Ray et al., 2019), a CRL benchmark with 3 tasks:

- *Goal*: In this task, the agent wins rewards by reaching the destinations (green cylinders) and gains cost for passing traps (blue circles) and hitting movable vases (cyan cubes).
- *Button*: In this task, the agent wins rewards by pushing a stationary button (orange balls) and gains cost for passing traps (blue circles) and hitting obstacles (purple cubes) with a fixed moving trajectory.
- *Push*: In this task, the agent wins rewards by pushing a crossing workpiece (a yellow column) to a specific destination (green cylinders) and gains cost for passing traps (blue circles) and hitting towers (Blue cylinders).

For each task, we have two difficulty levels, level-2 environments have more cost-consuming items than level-1 environments. Each environment is simulated in Mujoco (Todorov et al., 2012) with a *point* agent. To better explain the tasks, we provide screenshots for each environment in Fig.3.

We compare CONTROL to baselines from three domains: traditional policy-based RL methods (TRPO (Schulman et al., 2015),PPO Schulman et al. (2017)); Constrained RL methods (TRPO-Lagrangian, PPO-Lagrangian, CPO (Achiam et al., 2017)); Pareto approach(MGDA (Désidéri, 2012)). Moreover, we make ablation study by CONTROL-R (without gradient re-balancing) and CONTROL-C (without gradient clipping). We consider mean reward and mean cost as two metrics to weigh the effectiveness of models. For all methods we conduct 5 runs (1000 episodes each, 10000 steps each episode) with different random seeds and 5×100 episodes of test runs on another 5 random seeds. The threshold for cumulative cost is 25, as recommended in Ray et al. (2019). In order to show that our method is adaptive to various thresholds, we make another experiment with threshold 50.

For reproducibility, we list all architectures and hyper-parameters used in experiments in Appendix F.



(a) Goal-Lvl 1 **(b) Goal-Lvl 2** **(c) Button-Lvl 1** **(d) Button-Lvl 2** **(e) Push-Lvl 1** **(f) Push-Lvl 2**

Figure 3: Illustration of tasks in SafetyGym

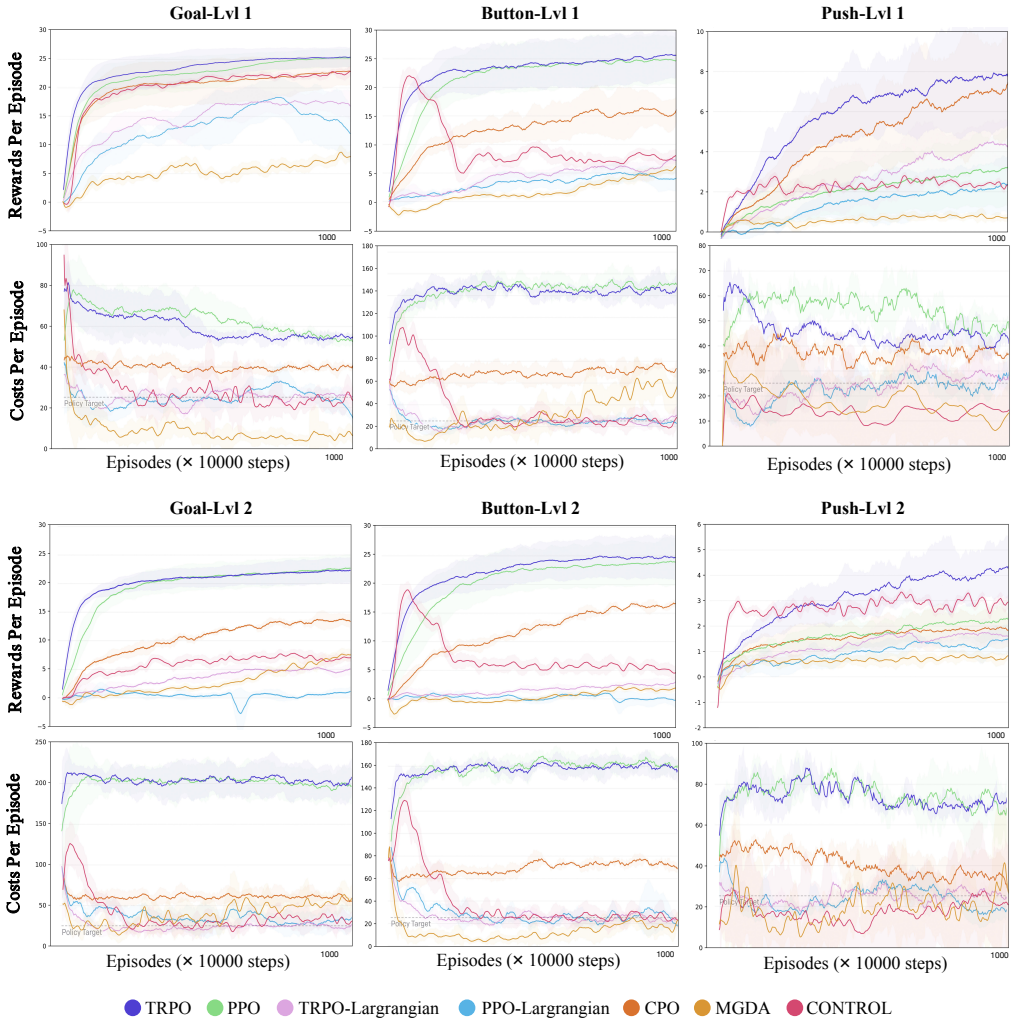


Figure 4: Reward and cost curves in all 6 tasks. A dashed line in cost curves represents the threshold. All lines are averaged over 5 runs and shaded areas indicate one standard deviation.

5.1 Experimental Analysis

Learning curves are provided in Fig. 4 and results of final tests are listed in Table. 1.

Comparison study In comparison, we prefer policies feasible to constraint, which means, any policy that fails to satisfy constraint is considered to be worse than any feasible policy. In all tasks, we can find a feasible policy even in all Lvl2 tasks, in which even Lagrangian methods sometimes fail to meet the threshold. Moreover, our method outperforms all baselines which find feasible policies in all tasks except *Push-Lvl1*. We argue that it is because our base model, PPO, also performs poorly in this task. In fact, all baselines learn limited experience in *push* tasks. By learning curves, we can notice that CONTROL's performance on reward improves quickly at the beginning of training and then decreases apparently. This decline indicates the effectiveness of gradient perturbation.

Baselines analysis **(i)** Lagrangian methods can find feasible or near-feasible policies in most tasks, but their performance on Lvl2 tasks are unsatisfactory **(ii)** CPO is fails to find feasible or even near-feasible policy, this is because reaching feasibility region in SafetyGym is challenging. **(iii)** As a deputy of existing Pareto approaches, MGDA's final policy has biased performance in reward and cost, which is either high reward and high cost or low reward and low cost.

Ablation study By comparing the performance of CONTROL and its variants, we can find that: **(i)** both gradient recalibration techniques is vital and effective for CONTROL **(ii)** the overall performance of CONTROL-P is better than CONTROL-R.

Adaptability analysis To demonstrate the adaptability of CONTROL, we compare it with Lagrangian methods under another threshold ($<$50). We can observe that our method shows consistent ability in satisfying predefined threshold and pursuing rewards.

6 Related Work

6.1 Constrained Reinforcement Learning

Constrained Reinforcement Learning is a generalized RL with regard to constraints in the environment. Conventionally, CRL is formulated as CMDP (Altman, 1999), in which the environment returns both a reward and non-negative costs state-wise. Such problems could be solved by Linear Programs when the set of states and actions are finite (see Chapter1.6 in Altman (1999)). But CMDPs with more complex environments are very tricky to handle.

CRL is broadly leveraged in several real-world applications, such as networks (Hou & Zhao, 2017), smart grids (Gao et al., 2020), and robotics (Dalal et al., 2018). Among all CRL scenarios, safety is the most common constraint. Safety CRL (Sui et al., 2015; Wachi et al., 2018) has more strict demand in constraints, which also raises the problem of safe exploration(Moldovan & Abbeel, 2012).

As aforementioned, mainstreams of the CRL literature are (i) Lagrangian methods (Borkar, 2005; Tessler et al., 2018; Stooke et al., 2020); (ii) Trust Region methods (Achiam et al., 2017; Yang et al., 2019). Besides, model-based CRL methods (Chow et al., 2017; Berkenkamp et al., 2017; Wachi & Sui, 2020) are worthy of being mentioned, which guarantee agents to explore in states with traceable from known low-cost states. Notably, Chow et al. (2018; 2019) utilizes Lyapunov functions techniques to analyze the stability of dynamical systems as safety constraints. However, most model-based CRL algorithms are restricted to discrete-action domains for their value-based modeling to environments.

6.2 Pareto Optimization

At present, Pareto optimizing (Fliege & Svaiter, 2000; Désidéri, 2012) provides a novel and time-economical way to solve multi-objective optimization problem by returning gradient descent directions that are beneficial to all objectives. This gradient is a linear combination of gradients of each objective, whose weights, called Pareto weight, are computed alongside the training process. Sener & Koltun (2018) first adapted the Pareto optimizer in Désidéri (2012) to deep learning by designing an approximate solver of Pareto weights. Similarly, Lin et al. (2019) improved Fliege & Svaiter (2000) in order to comply with multi-objective optimization problem with preference vector.

7 Conclusion

In this paper, we introduced a novel CRL paradigm named CONTROL from the perspective of Pareto optimization. The main challenges of applying existing Pareto approaches are imbalanced improvement over reward and cost and incapability of escaping from trivial Pareto-optimal policy. To overcome these challenges, we devise two gradient recalibration techniques, gradient re-balancing and gradient perturbation. To be specific, gradient re-balancing redefines original calculation method of Pareto weight and distributes more weights to underdeveloped objective; while gradient perturbation empowers us to temporarily sacrifice return to save costs when necessary. Experiments on a CRL benchmark, SafetyGym, validate the superiority of CONTROL and demonstrate a stable performance in satisfying constraint.

REFERENCES

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International Conference on Machine Learning*, pp. 22–31. PMLR, 2017.

Eitan Altman. *Constrained Markov decision processes*, volume 7. CRC Press, 1999.

Larry Armijo. Minimization of functions having lipschitz continuous first partial derivatives. *Pacific Journal of mathematics*, 16(1):1–3, 1966.

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

Vivek S Borkar. An actor-critic algorithm for constrained markov decision processes. *Systems & control letters*, 54(3):207–213, 2005.

Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International Conference on Machine Learning*, pp. 794–803. PMLR, 2018.

Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Research*, 18 (1):6070–6120, 2017.

Yinlam Chow, Ofir Nachum, Edgar A Duéñez-Guzmán, and Mohammad Ghavamzadeh. A lyapunov-based approach to safe reinforcement learning. In *NeurIPS*, 2018.

Yinlam Chow, Ofir Nachum, Aleksandra Faust, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. Lyapunov-based safe policy optimization for continuous control. In *International Conference on Learning Representations*, 2019.

Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe exploration in continuous action spaces. *arXiv preprint arXiv:1801.08757*, 2018.

Kalyanmoy Deb. Multi-objective optimization. In *Search methodologies*, pp. 403–449. Springer, 2014.

Jean-Antoine Désidéri. Multiple-gradient descent algorithm (mgda) for multiobjective optimization. *Comptes Rendus Mathematique*, 350(5-6):313–318, 2012.

Dongsheng Ding, Kaiqing Zhang, Tamer Basar, and Mihailo R Jovanovic. Natural policy gradient primal-dual method for constrained markov decision processes. In *NeurIPS*, 2020.

Jörg Fliege and Benar Fux Svaiter. Steepest descent methods for multicriteria optimization. *Mathematical methods of operations research*, 51(3):479–494, 2000.

Yuanqi Gao, Wei Wang, Jie Shi, and Nanpeng Yu. Batch-constrained reinforcement learning for dynamic distribution network reconfiguration. *IEEE Transactions on Smart Grid*, 11(6):5357–5369, 2020.

Chen Hou and Qianchuan Zhao. Optimization of web service-based control system for balance between network traffic and delay. *IEEE Transactions on Automation Science and Engineering*, 15(3):1152–1162, 2017.

Sham M Kakade. A natural policy gradient. *Advances in neural information processing systems*, 14, 2001.

Qi Kong, Liangliang Zhang, and Xin Xu. Constrained policy optimization algorithm for autonomous driving via reinforcement learning. In *2021 6th International Conference on Image, Vision and Computing (ICIVC)*, pp. 378–383. IEEE, 2021.

Xiao Lin, Hongjie Chen, Changhua Pei, Fei Sun, Xuanji Xiao, Hanxiao Sun, Yongfeng Zhang, Wenwu Ou, and Peng Jiang. A pareto-efficient algorithm for multiple objective optimization in e-commerce recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pp. 20–28, 2019.

Debabrata Mahapatra and Vaibhav Rajan. Multi-task learning with user preferences: Gradient descent with controlled ascent in pareto optimization. In *International Conference on Machine Learning*, pp. 6597–6607. PMLR, 2020.

Teodor Mihai Moldovan and Pieter Abbeel. Safe exploration in markov decision processes. In *ICML*, 2012.

Vilfredo Pareto. The new theories of economics. *Journal of political economy*, 5(4):485–502, 1897.

John C Platt and Alan H Barr. Constrained differential optimization. In *Proceedings of the 1987 International Conference on Neural Information Processing Systems*, pp. 612–621, 1987.

Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking safe exploration in deep reinforcement learning. *arXiv preprint arXiv:1910.01708*, 2019.

Harsh Satija, Philip Amortila, and Joelle Pineau. Constrained markov decision processes via backward value functions. In *International Conference on Machine Learning*, pp. 8502–8511. PMLR, 2020.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 525–536, 2018.

Adam Stooke, Joshua Achiam, and Pieter Abbeel. Responsive safety in reinforcement learning by pid lagrangian methods. In *International Conference on Machine Learning*, pp. 9133–9143. PMLR, 2020.

Yanan Sui, Alkis Gotovos, Joel Burdick, and Andreas Krause. Safe exploration for optimization with gaussian processes. In *International Conference on Machine Learning*, pp. 997–1005. PMLR, 2015.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 1998.

Chen Tessler, Daniel J Mankowitz, and Shie Mannor. Reward constrained policy optimization. In *International Conference on Learning Representations*, 2018.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012.

Akifumi Wachi and Yanan Sui. Safe reinforcement learning in constrained markov decision processes. In *International Conference on Machine Learning*, pp. 9797–9806. PMLR, 2020.

Akifumi Wachi, Yanan Sui, Yisong Yue, and Masahiro Ono. Safe exploration and optimization of constrained mdps using gaussian processes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Tsung-Yen Yang, Justinian Rosca, Karthik Narasimhan, and Peter J Ramadge. Projection-based constrained policy optimization. In *International Conference on Learning Representations*, 2019.

A ASSUMPTIONS

For the purpose of simplifying analysis, we make following assumptions:

Assumption 1. $\nabla_\omega J_R(\pi_\omega), \nabla_\omega J_C(\pi_\omega) \in \mathbb{R}^k$ *exist* $\forall \omega \in \mathbb{R}^k$.

Assumption 2. *The feasible policy set is not empty, i.e.* $\exists$ *a policy* π *s.t.* $J_C(\pi) \geq \zeta$.

Assumption 3. $\forall \omega * \in \mathbb{R}^k$ *which is a local optima in* $J_R(\pi_\omega)$*, exists a neighborhood* $U \subset \mathbb{R}^k$ *of* ω *s.t.* $\forall \omega' \in U, J_R(\pi_\omega) > J_R(\pi'_\omega)$.

Assumption 4. $\forall \omega * \in \mathbb{R}^k$ *which is a local optima in* $J_C(\pi_\omega)$*, exists a neighborhood* $U \subset \mathbb{R}^k$ *of* ω *s.t.* $\forall \omega' \in U, J_C(\pi_\omega) > J_C(\pi'_\omega)$.

Assumption 5. *An conservative policy* π_c *which yields no reward and cost always exist and is accessible.*

To analyze all assumptions, Assumption 1 is common in Deep Learning and guarantees the existence of gradient; Assumption 2 is the weakest prerequisite to ensure that a solution to a CMDP exists; Assumption 3 and Assumption 4 are requirements for completeness proof in Section B. Assumption 5 is common in many CMDPs, where a policy to stay and make no actions is π_c.

B PARETO-OPTIMAL SEARCHING IN CRL

In this section, we first re-elaborate the algorithms proposed in Fliege & Svaiter (2000) and Désidéri (2012)with concise proofs for their effectiveness under the settings of CRL problem. Furthermore, we show that those two methods are highly similar fundamentally. Notably, existing methods can only find Pareto-stationary policy, which would be defined later. **For completeness, we further prove that each Pareto-stationary policy is a Pareto-optimal policy with possibility 1 under Assumption 3,4 to CRL.**

Definition B.1 (Pareto-stationary Policy)**.** *For a policy* π_ω*,* π_ω *is Pareto stationary i.i.f.* $\nexists$ *a Pareto direction* $\mathbf{v}$ *of* π_ω *s.t.* $\langle \nabla_\omega J_R, v \rangle > 0$ *and* $\langle \nabla_\omega J_C, v \rangle > 0$.

As a necessary condition of Pareto-optimal, Pareto-stationary supplies a stronger condition for Pareto directions by strict inequations. Moreover, if a policy π_ω is not Pareto-stationary, $\nabla_\omega J_R(\pi_\omega), \nabla_\omega J_C(\pi_\omega) \neq \mathbf{0}$, otherwise $\langle \nabla_\omega J_R, v \rangle = 0$ or $\langle \nabla_\omega J_R, v \rangle = 0$ holds $\forall v \in \mathbb{R}^k$.

B.1 STEEPEST DESCENT METHOD (FLIEGE & SVAITER, 2000) IN CRL

As mentioned in Section 2, To search a Pareto optimal policy, the key is designing an iteration $\omega' = \omega + \eta(\omega)\Delta(w)$, where $\Delta(\omega)$ is update vector, and $\eta(\omega) \in \mathbb{R}_+$ is the stepsize. In Steepest Descent Methods (SDM), $\eta \in \mathbb{R}_+$ is obtained by line backtracking searching in a Armijo-Goldstein way(Armijo, 1966), we omit this part because it is not concerned in CONTROL. Now we are presenting how SDM searches $\Delta(\omega)$.

For a vector $\mathbf{v} \in \mathbb{R}^k$, define a function $f_\omega(\mathbf{v})$:

$$f_\omega(\mathbf{v}) := min(\langle \nabla_\omega J_R(\pi_\omega), \mathbf{v} \rangle, \langle \nabla_\omega J_C(\pi_\omega), \mathbf{v} \rangle). \tag{9}$$

Then $\Delta(\omega)$ is one of the solution of the problem below:

$$\max_{\mathbf{v}} \ f_\omega(\mathbf{v}) - \frac{1}{2}||\mathbf{v}||_2^2. \tag{10}$$

Theorem B.1. *If* π_ω *is Pareto-stationary, then* $\Delta(\omega)$ *is unique and* $\Delta(\omega) = \mathbf{0} \in \mathbb{R}^k$*. Otherwise, each* $\Delta(\omega)$ *is a Pareto direction of* π_ω*.*

Proof. As defined, $\Delta(\omega)$ is the solution of Problem(10). According to Definition B.1, if π_ω is Pareto-stationary, then $max_{\mathbf{v}} \ f_\omega(\mathbf{v}) \leq 0$. Thus:

$$\max_{\mathbf{v}} \ f_\omega(\mathbf{v}) - \frac{1}{2}||\mathbf{v}||_2^2 \leq \max_{\mathbf{v}} \ f_\omega(\mathbf{v}) + \max_{\mathbf{v}} \ (-\frac{1}{2}||\mathbf{v}||_2^2) \leq 0 + 0 = 0. \tag{11}$$

Note that when $\mathbf{v} = \mathbf{0}$, $f_\omega(\mathbf{v}) - \frac{1}{2}||\mathbf{v}||_2^2 = 0$. So $max_v\ f_\omega(\mathbf{v}) - \frac{1}{2}||\mathbf{v}||_2^2 = 0$. In addition, $\forall \mathbf{v} \neq \mathbf{0}, max_{\mathbf{v}}\ f_\omega(\mathbf{v}) - \frac{1}{2}||\mathbf{v}||_2^2 \leq max_{\mathbf{v}}\ -\frac{1}{2}||\mathbf{v}||_2^2 < 0$, so $\mathbf{0}$ is the only solution, i.e. $\Delta(\omega) = \mathbf{0}$.

Now let's consider the situation when π_ω is not Pareto-stationary, where $\exists \mathbf{v}\ s.t.\ f_\omega(\mathbf{v}) > 0$.

For a randomly selected $\delta \in (0, \frac{2f_\omega(\mathbf{v})}{||\mathbf{v}||_2^2})$:

$$f_\omega(\delta\mathbf{v}) - \frac{1}{2}||\delta\mathbf{v}||_2^2 = \delta(f_\omega(\mathbf{v}) - \frac{\delta}{2}||\mathbf{v}||_2^2) > 0$$

Thus:

$$f_\omega(\Delta(\omega)) - \frac{1}{2}||\Delta(\omega)||_2^2 = \max_{\mathbf{v}}\ f_\omega(\mathbf{v}) - \frac{1}{2}||\mathbf{v}||_2^2 > 0$$

Then $f_\omega(\Delta(\omega)) > \frac{1}{2}||\Delta(\omega)||_2^2 \geq 0$, which means $\langle \nabla_\omega J_R(\pi_\omega), \mathbf{v}\rangle, \langle \nabla_\omega J_C(\pi_\omega), \mathbf{v}\rangle > 0$. Thus, $\Delta(\omega)$ is a Pareto direction of π_ω by Definition 2.3. □

B.2 Multiple-gradient descent algorithm (Désidéri, 2012) in CRL

Multiple-gradient descent algorithm (MGDA) is introduced in Section 2.2, which attains Pareto weights by solving:

$$\begin{aligned} \min_{\beta_R, \beta_C \in \mathbb{R}} \quad & ||\beta_R \nabla_\omega J_R(\pi_\omega) + \beta_C \nabla_\omega J_C(\pi_\omega)||_2^2, \\ s.t. \quad & \beta_R, \beta_C \geq 0,\ \beta_R + \beta_C = 1. \end{aligned}$$

According to the Chapter.4 of Boyd et al. (2004), the Problem (4) is a convex optimization problem, so the solution $\beta^* = (\beta_R^*, \beta_C^*) \in \mathbb{R}^2$ exists. Furthermore, if $\nabla_\omega J_R(\pi_\omega), \nabla_\omega J_C(\pi_\omega) \neq \mathbf{0}$, then Problem (4) is strongly convex and β^* is unique.

After finding the solution of this problem, i.e. β^*, MGDA determine the Pareto direction $\Delta(\omega)$ as $\beta_R^* \nabla_\omega J_R(\pi_\omega) + \beta_C^* \nabla_\omega J_C(\pi_\omega)$.

Theorem B.2. *If $\pi_{\omega'}$ is a Pareto-stationary policy then $\Delta(\omega') = \mathbf{0}$.*

Proof. If $\nabla_{\omega'} J_R(\pi_{\omega'}) = \mathbf{0}$ or $\nabla_{\omega'} J_C(\pi_{\omega'}) = \mathbf{0}$, then $(1, 0)$ or $(0, 1)$ is a trivial solution of Problem (4).

Now let us discuss the situation that $\nabla_{\omega'} J_R(\pi_{\omega'}), \nabla_{\omega'} J_C(\pi_{\omega'}) \neq \mathbf{0}$. Under such conditions, β^* is unique. Since $\pi_{\omega'}$ is Pareto-stationary, by definition ω is one of the solution of the following problem:

$$\begin{aligned} \min_{\omega \in U} \quad & -J_{r_0}(\pi_\omega) \\ s.t. \quad & J_R(\pi_\omega) \geq J_R(\pi_{\omega'}), J_C(\pi_\omega) \geq J_C(\pi_{\omega'}) \end{aligned} \tag{12}$$

We can write the Lagrangian of Problem (12):

$$\mathcal{L}(\omega, \mu) = -J_R(\pi_\omega) - \mu_R \left[J_R(\pi_\omega) - J_R(\pi_{\omega'})\right] - \mu_C \left[J_C(\pi_\omega) - J_C(\pi_{\omega'})\right] \tag{13}$$

where $\mu = (\mu_R, \mu_C) \in \mathbb{R}_+^2$ is the vector of Lagrange multipliers. By Karush–Kuhn–Tucker(KKT) conditions, for a saddle point (ω', μ^*), where $\mu^*(\mu_R^*, \mu_C^*) \in \mathbb{R}_+^2$, we have:

$$\begin{aligned} & \frac{\partial \mathcal{L}}{\partial \omega}|_{\omega=\omega'} = \mathbf{0} \\ \Rightarrow & \nabla_{\omega'} J_R(\pi_{\omega'}) + \mu_R^* \nabla_{\omega'} J_R(\pi_{\omega'}) + \mu_C^* \nabla_{\omega'} J_C(\pi_{\omega'}) = \mathbf{0} \\ \Rightarrow & \frac{1 + \mu_C^*}{1 + \mu_R^* + \mu_C^*} \nabla_{\omega'} J_R(\pi_{\omega'}) + \sum_{j=1}^{n} \frac{\mu_C^*}{1 + \mu_R^* + \mu_C^*} \nabla_{\omega'} J_C(\pi_{\omega'}) = \mathbf{0} \end{aligned} \tag{14}$$

As $||\cdot||_2^2 \geq 0$ and the uniqueness of β^*, $(\frac{1+\mu_C^*}{1+\mu_R^*+\mu_C^*}, \frac{\mu_C^*}{1+\mu_R^*+\mu_C^*})$ is the only solution of Problem (14). Then $\Delta(\omega') = \frac{1+\mu_C^*}{1+\mu_R^*+\mu_C^*} \nabla_{\omega'} J_R(\pi_{\omega'}) + \frac{\mu_C^*}{1+\mu_R^*+\mu_C^*} \nabla_{\omega'} J_C(\pi_{\omega'}) = 0$.

□

Theorem B.3. *If $\pi_{\omega'}$ is not a Pareto stationary policy, then $\Delta(\omega') \neq \mathbf{0}$ and $\Delta(\omega')$ is a Pareto direction of $\pi_{\omega'}$.*

Proof. We first prove that $\Delta(\omega') \neq \mathbf{0}$. By definition, $\exists v \in \mathbb{R}^k$ s.t.$\langle \nabla_\omega J_R, v\rangle > 0, \langle \nabla_\omega J_C, v\rangle > 0$. So $\langle v, \Delta(\omega')\rangle = \beta_R^* \langle \nabla_{\omega'} J_R(\pi_{\omega'}), v\rangle > 0 + \beta_C^* \langle \nabla_{\omega'} J_C(\pi_{\omega'}), v\rangle > 0$, which implies $\Delta(\omega') \neq \mathbf{0}$.

The Lagrangian of the Problem 4 is :

$$\mathcal{L}(\beta, \lambda, \mu) = ||\beta_R \nabla_{\omega'} J_R(\pi'_\omega) + \beta_C \nabla_{\omega'} J_C(\pi'_\omega)||_2^2 + \lambda(\beta_R + \beta_C - 1) - \mu_R \beta_R - \mu_C \beta_C$$

where $\lambda, \mu = (\mu_R, \mu_C)$ are Lagrange multipliers and $\mu_R, \mu_C \geq 0, \mu_R \beta_R, \mu_C \beta_C = 0$.

Say $(\beta^*, \lambda^*, \mu^*)$ is a saddle point of above problem, by KKT conditions, we have:

$$\begin{cases} 2\Delta(\omega') \cdot \nabla_{\omega'} J_R(\pi_{\omega'}) + \lambda^* - \mu_R^* = 0, \\ 2\Delta(\omega') \cdot \nabla_{\omega'} J_C(\pi_{\omega'}) + \lambda^* - \mu_C^* = 0, \\ \mu_R^*, \mu_C^* \geq 0, \mu_R^* \beta_R^* = \mu_C^* \beta_C^* = 0. \end{cases}$$

Multiply the first two equations with corresponding β_R, β_C and sum:

$$2\Delta(\omega') \cdot \Delta(\omega') + \lambda(\beta_R^* + \beta_C^*) - \mu_R^* \beta_R^* - \mu_C^* \beta_C^* = 0.$$

Since $\mu_i^* \beta_i^* = 0$ and $\beta_R^* + \beta_C^* = 1$, we have $\lambda^* = -2||\Delta(\omega')||_2^2$. Note that $\Delta(\omega') \neq \mathbf{0}$, so $\lambda^* < 0$. Thus:

$$\begin{aligned} \langle \Delta(\omega'), \nabla_{\omega'} J_R(\pi_{\omega'})\rangle &= \mu_R^* - \lambda^* \geq -\lambda^* > 0, \\ \langle \Delta(\omega'), \nabla_{\omega'} J_C(\pi_{\omega'})\rangle &= \mu_C^* - \lambda^* \geq -\lambda^* > 0, \end{aligned} \tag{15}$$

By definition, $\Delta_{\omega'}$ is a Pareto direction of $\pi_{\omega'}$. □

MGDA could be very efficiency because Problem (4) has explicit solution:

$$\beta_R^* = \begin{cases} 0, & \beta_0 < 0 \\ \beta_0, & 0 \leq \beta_0 \leq 1 \\ 1, & \beta_0 > 1 \end{cases}, \quad \beta_C^* = 1 - \beta_R^*, \tag{16}$$

where $\beta_0 = \dfrac{\nabla_\omega J_C(\pi_\omega) \cdot (\nabla_\omega J_C(\pi_\omega) - \nabla_\omega J_R(\pi_\omega))}{||\nabla_\omega J_C(\pi_\omega) - \nabla_\omega J_R(\pi_\omega)||_2^2}$.

B.3 Discussion of existing methods

SDM and MGDA are basically solving the same optimization problem. Actually, if we transform the Problem (10) into a constrained optimization problem:

$$\begin{aligned} \max_{\mathbf{v}} \quad & \alpha - \frac{1}{2}||\mathbf{v}||_2^2 \\ s.t. \quad & \langle \nabla_\omega J_R(\pi_\omega), \mathbf{v}\rangle \geq \alpha, \langle \nabla_\omega J_C(\pi_\omega), \mathbf{v}\rangle \geq \alpha. \end{aligned} \tag{17}$$

It is easy to prove that Problem (17) and Problem (4) are primal-dual problem. As the Slater condition holds, these two optimization will yield identical optima (Boyd et al., 2004).

In addition, Lemma 1 in Fliege & Svaiter (2000) also points that the Pareto direction $\Delta(\omega)$ derived from Problem (17) and Problem (4) is the steepest gradient in MOOP settings.

In the body part, we choose MGDA to represent Pareto-optimal searching algorithm. Because MGDA is more efficient and comprehensible.

B.4 COMPLETENESS PROOF

So far we can only search Pareto-stationary policy instead of Pareto-optimal. To fill the gap, we make some proof for completeness.

Lemma B.4. *For a policy π_ω, if $\nabla_\omega J_R(\pi_\omega)$ or $\nabla_\omega J_C(\pi_\omega)$ is $\mathbf{0}$, then π_ω is Pareto-optimal.*

Proof. If $\nabla_\omega J_R(\pi_\omega) = \mathbf{0}$, then ω is a local optima as for J_{π_ω}. According to Assumption 4, there is a neighborhood $U \subset \mathbb{R}^k$ of ω s.t. $\forall \omega' \in U, J_C(\pi_\omega) > J_C(\pi'_\omega)$. Then π_ω is Pareto-optimal since $\forall \omega' \in U J_C(\pi'_\omega) \not\succ J_C(\pi_\omega)$. It is similar to the case of $\nabla_\omega J_C(\pi_\omega) = \mathbf{0}$. □

Theorem B.5. *If a policy π_ω is Pareto-stationary, it is Pareto-optimal with possibility 1.*

Proof. Suppose we have a policy $\pi_{\omega'}$ which is Pareto-stationary but not Pareto-optimal. Then, by Lemma B.4 we know $\nabla_\omega J_R(\pi_\omega), \nabla_\omega J_C(\pi_\omega) = \mathbf{0}$, otherwise $\pi_{\omega'}$ is Pareto-optimal.

By solving Problem (4), we can obtain a Pareto direction $\Delta(\omega')$. By Theorem (B.2), $\Delta(\omega') = \mathbf{0}$, which implies that $\nabla_\omega J_R(\pi_\omega), \nabla_\omega J_C(\pi_\omega)$ are co-linear.

Note that the reward and cost are independent, which means we can regard $\nabla_\omega J_R(\pi_\omega), \nabla_\omega J_C(\pi_\omega)$ as random vectors with regard to ω. As our model has many parameters, i.e. $k >> 2$, the possibility that $\nabla_\omega J_R(\pi_\omega), \nabla_\omega J_C(\pi_\omega)$ are co-linear is 0.

Thus, a Pareto-stationary $\pi_{\omega'}$ is Pareto-optimal with possibility 1. □

C CONNECTIONS TO PRIOR WORKS

In this section, we prove that existing CRL algorithms are fundamentally searching Pareto optimal policy. In other words, if these algorithms converge at a policy π_ω, then π_ω is Pareto-optimal, or even Pareto-feasible when constraint is satisfied in practical. The proofs below follow the notation and problem definition in Section 2.

C.1 LINEAR SCALARIZATION

Linear scalarization method aggregate all optimizing targets into one by linear summation with predefined weight vector $\lambda = (\lambda_R, \lambda_C) \in \mathbb{R}^2_+$. So now the optimizing problem is:

$$\max_{\omega} \quad \lambda_R J_R(\pi_\omega) + \lambda_C J_C(\pi_\omega). \tag{18}$$

Proposition C.1. *If ω^* is one of a local optima of above problem, then π_{ω^*} is Pareto optimal.*

Proof. Assume π_{ω^*} is one of the local optima of Problem (18) but not Pareto optimal, then $\exists \omega' \in U$ s.t. $\pi_{\omega'} \succ \pi_{\omega^*}$, where $U \in \mathbb{R}^k$ is a neighborhood of ω^*. Then, $\lambda_R J_R(\pi_{\omega'}) + \lambda_C J_C(\pi_{\omega'}) > \lambda_R J_R(\pi_{\omega^*}) + \lambda_C J_C(\pi_{\omega^*})$, which is contradict with the original assumption. Thus, the original proposition holds. □

This proposition suggests that accessing a Pareto optimal policy is simple and feasible. However, the policy generated by linear scalarization is strongly related to the choose of λ_R, λ_C and may not satisfies the constraint.

C.2 LAGRANGIAN METHODS

Lagrangian methods (Tessler et al., 2018; Ray et al., 2019) transform a constrained optimization problem into a normal min-max optimization problem by adding Lagrange multipliers $\lambda \in \mathbb{R}_+$:

$$\begin{aligned} \max_{\pi} \min_{\lambda \geq 0} \quad & \mathcal{L}(\pi, \lambda) := J_R(\pi) + \lambda \cdot (J_C(\pi) - \zeta), \\ s.t. \quad & J_C(\pi) \geq \zeta. \end{aligned} \tag{19}$$

To solve this problem, we can apply gradient ascend on π 's parameters and gradient descent on λ. of which the convergence proof is in (Platt & Barr, 1987). Similarly, we have:

Proposition C.2. *If π^*, λ^* is one of the solution of Problem (19), then π^* is Pareto-optimal.*

Proof. Assume $(\pi_{\omega^*}, \lambda^*)$ is one of the saddle points of above problem but π_{ω^*} is not Pareto optimal, then $\exists \omega' \in U$ s.t. $\pi_{\omega'} \succ \pi_{\omega^*}$, where $U \in \mathbb{R}^k$ is a neighborhood of ω^*. Then:

$$J_R(\pi_{\omega'}) + \lambda^* \cdot (J_C(\pi_{\omega'}) - \zeta) \geq J_R(\pi_{\omega^*}) + \lambda \cdot (J_C(\pi_{\omega^*}) - \zeta).$$

This suggests that $(\pi_{\omega^*}, \lambda^*)$ is not a saddle point, which is contradict with original assumption. □

Though provide a strong reliable guarantee to satisfy constraint, Lagrangian methods may be too conservative to explore effectively. This is because λ would control the optimization when $(J_C(\pi) - \zeta)$ is big, which is common at the beginning of the CRL training session.

C.3 Constrained Policy optimization (Achiam et al., 2017)

Constrained Policy optimization (CPO) is a CRL algorithm by updating policy within a constraint satisfying region. It updates policy by solving:

$$\begin{aligned} \pi_{k+1} = &\arg\max_{\pi} \mathbb{E}_{s\sim\pi_k, a\sim\pi}\left[A_R^{\pi_k}(s,a)\right] \\ s.t. \quad &J_{C_i}(\pi_k) - \mathbb{E}_{s\sim\pi_k, a\sim\pi}\left[A_C^{\pi_k}(s,a)\right] \geq \zeta, \\ &\bar{D}_{KL}(\pi \| \pi_k) \leq \delta. \end{aligned} \tag{20}$$

To prove that CPO is searching Pareto-optimal policy, we need some basic conclusion in RL:

Lemma C.1 (Kakade (2001))**.** *Given an existing policy π_{old}, then:*

$$\begin{aligned} J_R(\pi) - J_R(\pi_{old}) &= \mathbb{E}_{s\sim\pi, a\sim\pi}\left[A_R^{\pi_{\text{old}}}(s,a)\right], \\ J_C(\pi) - J_C(\pi_{old}) &= \mathbb{E}_{s\sim\pi, a\sim\pi}\left[-A_C^{\pi_{\text{old}}}(s,a)\right]. \end{aligned}$$

Lemma C.2 (Theorem 1 in Schulman et al. (2015))**.** *Let $\epsilon = max_s |\mathbb{E}_{a\sim\pi}\left[A^{\pi_{old}}(s,a)\right]|$,then:*

$$\begin{aligned} \mathbb{E}_{s\sim\pi, a\sim\pi}\left[A_R^{\pi_{\text{old}}}(s,a)\right] &> \mathbb{E}_{s\sim\pi_{old}, a\sim\pi}\left[A_R^{\pi_{\text{old}}}(s,a)\right] - C \max_s D_{\text{KL}}^s(\pi_{old}, \pi), \\ \mathbb{E}_{s\sim\pi, a\sim\pi}\left[A_C^{\pi_{\text{old}}}(s,a)\right] &> \mathbb{E}_{s\sim\pi_{old}, a\sim\pi}\left[A_C^{\pi_{\text{old}}}(s,a)\right] - C \max_s D_{\text{KL}}^s(\pi_{old}, \pi). \end{aligned}$$

where $C = 4\epsilon\gamma/(1-\gamma)^2$ and D_{KL}^s denotes the Kullback–Leibler divergence of two policies when making decisions in state s.

Proposition C.3. *If CPO converges at π_{ω^*}, then π_{ω^*} is Pareto-optimal.*

Proof. Suppose π_{ω^*} is Pareto-optimal is not Pareto-optimal, then $\exists \omega' \in U$ s.t. $\pi_{\omega'} \succ \pi_{\omega^*}$, where $U \in \mathbb{R}^k$ is a neighborhood of ω^*.

The original Problem (20) can absorb one constraint, and transform into:

$$\begin{aligned} \pi_{k+1} = &\arg\max_{\pi} \mathbb{E}_{s\sim\pi_k, a\sim\pi}\left[A_R^{\pi_k}(s,a)\right] - C \max_s D_{\text{KL}}^s(\pi_k, \pi), \\ s.t. \quad &J_{C_i}(\pi_k) - \mathbb{E}_{s\sim\pi_k, a\sim\pi}\left[A_C^{\pi_k}(s,a)\right] \geq \zeta, \end{aligned} \tag{21}$$

where the C is consistent with Lemma C.2. As CPO converges at π_{ω^*}, so:

$$\pi_{\omega^*} = \arg\max_{\pi} \mathbb{E}_{s\sim\pi_{\omega^*}, a\sim\pi}\left[A_R^{\pi_{\omega^*}}(s,a)\right] - C \max_s D_{\text{KL}}^s(\pi_{\omega^*}, \pi). \tag{22}$$

Since $\mathbb{E}_{s\sim\pi_{\omega^*}, a\sim\pi_{\omega^*}}\left[A_R^{\pi_{\omega^*}}(s,a)\right] - C\, max_s\, D_{\text{KL}}^s(\pi_{\omega^*}, \pi_{\omega^*}) = 0$, then:

$$\mathbb{E}_{s\sim\pi_{\omega^*}, a\sim\pi_{\omega'}}\left[A_R^{\pi_{\omega^*}}(s,a)\right] - C \max_s D_{\text{KL}}^s(\pi_{\omega^*}, \pi_{\omega'}) \leq 0. \tag{23}$$

So we have:

$$\begin{aligned} &J_R(\pi_{\omega'}) - J_R(\pi_{\omega^*}) \\ =&\mathbb{E}_{s\sim\pi_{\omega'}, a\sim\pi_{\omega'}}\left[A_R^{\pi_{\omega^*}}(s,a)\right] && \text{Lemma C.1} \\ <&-\mathbb{E}_{s\sim\pi_{\omega'}, a\sim\pi_{\omega^*}}\left[A_R^{\pi_{\omega'}}(s,a)\right] + C \max_s D_{\text{KL}}^s(\pi_{\omega'}, \pi_{\omega^*}) && \text{Lemma C.2} \\ =&\mathbb{E}_{s\sim\pi_{\omega^*}, a\sim\pi_{\omega'}}\left[A_R^{\pi_{\omega^*}}(s,a)\right] - C \max_s D_{\text{KL}}^s(\pi_{\omega^*}, \pi_{\omega'}) \leq 0 && \text{Formula (23)} \end{aligned} \tag{24}$$

The Formula (24) is contradict to $\pi_{\omega'} \succ \pi_{omega^*}$, thus π_{ω^*} is Pareto-optimal. □

D PROOFS ABOUT CONTROL

D.1 PROOFS FOR GRADIENT NORMALIZATION

Lemma 4.1. *If π_ω is not Pareto-optimal, $\Delta_N(\omega) := \beta_R^N \nabla_\omega J_R(\pi_\omega) + \beta_C^N \nabla_\omega J_C(\pi_\omega)$ is a Pareto direction of π_ω.*

Proof. As a Pareto-optimal policy, π_ω is Pareto-stationary. By Theorem B.3, we know that $\langle \Delta_N(\omega), \nabla_\omega^N J_R(\pi_\omega)\rangle > 0$ and $\langle \Delta_N(\omega), \nabla_\omega^N J_C(\pi_\omega)\rangle > 0$.

Thus, $\langle \Delta_N(\omega), \nabla_\omega^N J_R(\pi_\omega)\rangle > 0$ and $\langle \Delta_N(\omega), \nabla_\omega^N J_C(\pi_\omega)\rangle > 0$. By definition, $\Delta_N(\omega)$ is a Pareto direction of π_ω. □

Before proceeding to prove Theorem 4.2, we need a lemma to estimate $J_R(\pi_{\omega'}) - J_R(\pi_\omega)$ and $J_C(\pi_{\omega'}) - J_C(\pi_\omega)$.

Lemma D.1. *With the iteration paradigm $\omega' = \omega + \eta(\omega)\Delta(\omega)$, if $\eta(\omega) \to 0$, then $J_R(\pi_{\omega'}) - J_R(\pi_\omega) = \eta(\omega)\langle \Delta(\omega), \nabla_\omega J_R(\pi_\omega)\rangle$ and $J_C(\pi_{\omega'}) - J_C(\pi_\omega) = \eta(\omega)\langle \Delta(\omega), \nabla_\omega J_C(\pi_\omega)\rangle$.*

Proof. We only need to prove one of $J_R(\pi_\omega), J_C(\pi_\omega)$, because the other one can be proved in a similar way. Make a first-order Taylor expansion of $J_R(\pi_\omega)$ at ω^*:

$$J_R(\pi_{\omega'}) = J_R(\pi_\omega) + (\omega' - \omega) \cdot \nabla_\omega J_R(\pi_\omega) + O\left[(\omega' - \omega)^2\right],$$

where $O\left[(\omega' - \omega)^2\right] \to 0$ if $\omega' - \omega \to \mathbf{0}$. Apply Gram-Schmidt orthogonalization to $\Delta(\omega)$, we have $\Delta(\omega) = \dfrac{\langle \Delta(\omega), \nabla_\omega J_R(\pi_\omega)\rangle}{||\nabla_\omega J_R(\pi_\omega)||_2 ||\Delta(\omega)||_2} \nabla_\omega J_R(\pi_\omega) + c\nabla_R^\perp(\omega)$, where c is a coefficient we are not interested and $\nabla_R^\perp(\omega)$ is a vector orthogonal to $\nabla_\omega J_R(\pi_\omega)$. Thus:

$$\begin{aligned} J_R(\pi_{\omega'}) - J_R(\pi_\omega) &= +(\omega' - \omega) \cdot \nabla_\omega J_R(\pi_\omega) + O\left[(\omega' - \omega)^2\right] \\ &= \eta(\omega) \left[\frac{\langle \Delta(\omega), \nabla_\omega J_R(\pi_\omega)\rangle}{||\nabla_\omega J_R(\pi_\omega)||_2^2} \nabla_\omega J_R(\pi_\omega) + c\nabla_R^\perp(\omega) \right] \cdot \nabla_\omega J_R(\pi_\omega) \\ &\quad + O\left[(\omega' - \omega)^2\right] \\ &= \eta(\omega)\langle \Delta(\omega), \nabla_\omega J_R(\pi_\omega)\rangle + O\left[(\omega' - \omega)^2\right]. \end{aligned} \tag{25}$$

Since $\eta(\omega) \to 0 \Rightarrow \omega' - \omega \to 0$, then we have: $J_R(\pi_{\omega'}) - J_R(\pi_\omega) = \eta(\omega)\langle \Delta(\omega), \nabla_\omega J_R(\pi_\omega)\rangle$. Similarly, we can infer $J_C(\pi_{\omega'}) - J_C(\pi_\omega) = \eta(\omega)\langle \Delta(\omega), \nabla_\omega J_C(\pi_\omega)\rangle$. □

Theorem 4.2 *Suppose the iteration paradigm is $\omega' = \omega + \eta(\omega)\Delta_N(\omega)$ and $\eta(\omega) \to 0$. Then:*
(i) *if we use $\Delta(\omega)$ derived from Problem (4), the improvements in reward and cost are similar. Specifically, when $\beta_R^* \in (0, 1)$, $\dfrac{J_R(\pi_{\omega'}) - J_R(\pi_\omega)}{J_C(\pi_{\omega'}) - J_C(\pi_\omega)} \to 1$.* ***(ii)*** *if we use $\Delta_N(\omega)$ derived from Problem (5), the improvements in reward and cost are proportional to the square length of corresponding gradient. Specifically, when $\beta_R^N \in (0, 1)$, $\dfrac{J_R(\pi_{\omega'}) - J_R(\pi_\omega)}{J_C(\pi_{\omega'}) - J_C(\pi_\omega)} \to \dfrac{||\nabla_\omega J_R(\pi_\omega)||_2^2}{||\nabla_\omega J_C(\pi_\omega)||_2^2}$.*

Proof. By Lemma D.1, we have:

$$\frac{J_R(\pi_{\omega'}) - J_R(\pi_\omega)}{J_C(\pi_{\omega'}) - J_C(\pi_\omega)} \to \frac{\eta(\omega)\langle \Delta(\omega), \nabla_\omega J_R(\pi_\omega)\rangle}{\eta(\omega)\langle \Delta(\omega), \nabla_\omega J_C(\pi_\omega)\rangle} = \frac{\langle \Delta(\omega), \nabla_\omega J_R(\pi_\omega)\rangle}{\langle \Delta(\omega), \nabla_\omega J_C(\pi_\omega)\rangle}, \tag{26}$$

(i) when $\beta_R^* \in (0, 1)$, $\Delta(\omega)$ is perpendicular to $\nabla_\omega J_R(\pi_\omega) - \nabla_\omega J_C(\pi_\omega)$, which indicates that $\langle \Delta(\omega), \nabla_\omega J_R(\pi_\omega) - \nabla_\omega J_C(\pi_\omega)\rangle = 0$. Thus:

$$\frac{J_R(\pi_{\omega'}) - J_R(\pi_\omega)}{J_C(\pi_{\omega'}) - J_C(\pi_\omega)} \to \frac{\langle \Delta(\omega), \nabla_\omega J_R(\pi_\omega)\rangle}{\langle \Delta(\omega), \nabla_\omega J_C(\pi_\omega)\rangle} = 1. \tag{27}$$

(ii) when $\beta_R^N \in (0,1)$, $\Delta(\omega)$ is perpendicular to $\nabla_\omega^N J_R(\pi_\omega) - \nabla_\omega^N J_C(\pi_\omega)$, which indicates that $\langle \Delta(\omega), \nabla_\omega^N J_R(\pi_\omega) - \nabla_\omega^N J_C(\pi_\omega) \rangle = 0$. Thus:

$$\frac{J_R(\pi_{\omega'}) - J_R(\pi_\omega)}{J_C(\pi_{\omega'}) - J_C(\pi_\omega)} \to \frac{\langle \Delta(\omega), \nabla_\omega^N J_R(\pi_\omega) \rangle}{\langle \Delta(\omega), \nabla_\omega^N J_C(\pi_\omega) \rangle} = \frac{||\nabla_\omega J_R(\pi_\omega)||_2^2 \langle \Delta(\omega), \nabla_\omega J_R(\pi_\omega) \rangle}{||\nabla_\omega J_C(\pi_\omega)||_2^2 \langle \Delta(\omega), \nabla_\omega J_C(\pi_\omega) \rangle} = \frac{||\nabla_\omega J_R(\pi_\omega)||_2^2}{||\nabla_\omega J_C(\pi_\omega)||_2^2}. \tag{28}$$

□

D.2 Proofs for Gradient Perturbation

Lemma D.2 (Theorem.1 in Schulman et al. (2015)). *Let $\epsilon = max_s |\mathbb{E}_{a\sim\pi_{\omega'}} [A^{\pi_\omega}(s,a)]|$, then:*

$$J_C(\pi_{\omega'}) - J_C(\pi_\omega) \geq \mathbb{E}_{s\sim\pi_\omega, a\sim\pi_{\omega'}} \left[-A_C^{\pi_\omega}(s_t, a_t)\right] - C D_{\mathrm{KL}}^{\max}(\pi_{\omega'}, \pi_\omega), \tag{29}$$

where $C = 4\epsilon\gamma/(1-\gamma)^2$.

Theorem 4.3 (The lower bound of $J_C(\pi)$ improvement) *Given the parameter updating paradigm $\omega' = \omega + \eta(\omega)\Delta_N(\omega)$ and clipping threshold t, we have the lower bound for $J_C(\pi'_\omega) - J_C(\pi_\omega)$:*

$$\begin{aligned} &J_C(\pi_{\omega'}) - J_C(\pi_\omega) \geq \\ &\eta(\omega)\left[t||\nabla_\omega J_C(\pi_\omega)||_2^2 \langle \nabla_\omega^N J_R(\pi_\omega) - \nabla_\omega^N J_C(\pi_\omega), \nabla_\omega^N J_C(\pi_\omega) \rangle + 1\right] - C D_{\mathrm{KL}}^{\max}(\pi_{\omega'}, \pi_\omega), \end{aligned} \tag{30}$$

where $C = 4\epsilon\gamma/(1-\gamma)^2$ and $\epsilon = max_s |\mathbb{E}_{a\sim\pi_{\omega'}} [A^{\pi_\omega}(s,a)]|$. And this bound is strictly positive related to t.

Proof. By Lemma D.2,

$$\begin{aligned} &J_C(\pi_{\omega'}) - J_C(\pi_\omega) \geq \\ &\mathbb{E}_{s\sim\pi_\omega, a\sim\pi_{\omega'}} \left[-A_C^{\pi_\omega}(s_t, a_t)\right] - \mathbb{E}_{s\sim\pi_{\omega'}, a\sim\pi_{\omega'}} \left[-A_C^{\pi_\omega}(s_t, a_t)\right] - C D_{\mathrm{KL}}^{\max}(\pi_{\omega'}, \pi_\omega). \end{aligned} \tag{31}$$

The C in last inequation is consistent with the setting in Lemma D.2. Since $\nabla_\omega J_C(\pi_\omega)$ is differentiated from $\mathbb{E}_{s\sim\pi_\omega, a\sim\pi_{\omega'}} \left[-A_C^{\pi_\omega}(s_t, a_t)\right]$, consider Lemma D.1, then:

$$\begin{aligned} &J_C(\pi_{\omega'}) - J_C(\pi_\omega) \\ \geq &\eta(\omega)\langle \Delta_N(\omega), \nabla_\omega J_C(\pi_\omega) \rangle - C D_{\mathrm{KL}}^{\max}(\pi_{\omega'}, \pi_\omega) \\ = &\eta(\omega)\langle min(t, \beta_R^N)\nabla_\omega^N J_R(\pi_\omega) + \left[1 - min(t, \beta_R^N)\right] \nabla_\omega^N J_C(\pi_\omega), \nabla_\omega J_C(\pi_\omega) \rangle - C D_{\mathrm{KL}}^{\max}(\pi_{\omega'}, \pi_\omega) \\ \geq &\eta(\omega)\langle t\nabla_\omega J_R(\pi_\omega) + (1-t)\nabla_\omega J_C(\pi_\omega), \nabla_\omega J_C(\pi_\omega) \rangle - C D_{\mathrm{KL}}^{\max}(\pi_{\omega'}, \pi_\omega) \\ = &\eta(\omega)\left[t||\nabla_\omega J_C(\pi_\omega)||_2^2 \langle \nabla_\omega^N J_R(\pi_\omega) - \nabla_\omega^N J_C(\pi_\omega), \nabla_\omega^N J_C(\pi_\omega) \rangle + 1\right] - C D_{\mathrm{KL}}^{\max}(\pi_{\omega'}, \pi_\omega) \end{aligned} \tag{32}$$

This bound is sensitive to both a and $\eta(\omega)$, and it strictly positive correlated with t, because according to Eq (16), $\langle \nabla_\omega^N J_R(\pi_\omega) - \nabla_\omega^N J_C(\pi_\omega), \nabla_\omega^N J_C(\pi_\omega) \rangle$ is strictly positive. Otherwise, $\beta_R^N = 0$, the clipping is not working. □

This lower bound is consistent with the lower bound when clipping is not working (i.e. Formula (D.2)) and tighter when clipping is working (i.e. $\beta_R^N > t$) and we perturb the $\Delta_N(\omega)$ to bias to $\nabla_\omega J_C(\pi_\omega)$.

E PSEUDO-CODE OF CONTROL

Algorithm 1: CONTROL

input : Threshold c_0 for cumulative cost, initial clipping threshold t_0, mean cumulative reward of last epoch r_{LE}, mean cumulative cost of last epoch c_{LE}

Initialize actor parameters ω, cost critic θ_C, reward critic θ_R randomly;
Initialize $t_0 \to 0.5$, $r_{LE} \to 0$, $c_{LE} \to 0$ **for** $B = 0, 1, ...$ **do**
  for $E = 0, 1, ...N-1$ **do**
    Sample d episodes $\{\tau_1, \tau_2, ...\tau_d\}, \tau_i \sim \pi_\omega$;
    Record mean cumulative reward and cost $\hat{r}, \hat{c}$ in $\{\tau_1, \tau_2, ...\tau_d\}$;
    Update θ_R, θ_C by MSE loss;
    Obtain $\nabla_\omega J_R(\pi_\omega), \nabla_\omega J_C(\pi_\omega)$ by Eq (8);
    Obtain β_R^N, β_C^N by solving Problem (5);
    if $\hat{c} > c_0$ *and* $\hat{c} > c_{LE}$; // Constraint is not satisfied and $\hat{c}$ is worse
    then
      $\beta_R^N \to min(t, \beta_R^N), \beta_C^N \to 1 - \beta_R^N$; // Clip β_R^N
      Update ω with gradient $\beta_R^N \nabla_\omega J_R(\pi_\omega) + \beta_C^N \nabla_\omega J_C(\pi_\omega)$;
      $t_0 \to t_0 - t'$; // Decrease clipping threshold
      $r_{LE} \to \hat{r}$, $c_{LE} \to \hat{c}$
    else if $\hat{c} < c_0$ *and* $c_{LE} < c_0$ *and* $\hat{r} < r_{LE}$; // Constraint is satisfied in a row and $\hat{r}$ is worse
    then
      $\beta_C^N \to min(t, \beta_C^N), \beta_R^N \to 1 - \beta_C^N$; // Clip β_C^N
      Update ω with gradient $\beta_R^N \nabla_\omega J_R(\pi_\omega) + \beta_C^N \nabla_\omega J_C(\pi_\omega)$;
      $t_0 \to t_0 + t'$; // Increase clipping threshold
      $r_{LE} \to \hat{r}$, $c_{LE} \to \hat{c}$
    else
      Update ω with gradient $\beta_R^N \nabla_\omega J_R(\pi_\omega) + \beta_C^N \nabla_\omega J_C(\pi_\omega)$; // No clipping
      $r_{LE} \to \hat{r}$, $c_{LE} \to \hat{c}$
    end
  end
end
return: A policy π_ω

F EXPERIMENT DETAILS

F.1 ARCHITECTURE DETAILS

CONTROL is built on an Actor-Critic framework. To model state value functions with regard to rewards and costs separately, we adopt two critics. For both actor and critic networks, we use identical structure, which is an MLP with a 64-dimension hidden layer. Except the output layer, we apply $tanh()$ as the activation function. For all code-level implementation we used PyTorch.

F.2 HYPER-PARAMETERS

Model training For learning rate, we choose 0.0012 for the actor and 0.001 for the critics. Once sample enough episodes, we reuse the buffers for 50 times. To adapt MDPs in SafetyGym, which is with continuous states and actions, the output of the actor network is all means of a Multivariate Gaussian distribution, and the standard deviation is locked as 0.6. While in testing, we resize all standard deviation to 0.4 to low down the agent's desire to explore. Besides, we apply GAE (Schulman et al., 2016) to estimate advantage with $\gamma = 0.99$ and $\lambda = 0.95$.

Clipping threshold evolving We initialize the clipping threshold t_0 as 0.5, and check if we should update it every 100000 steps. If so, we increase or decrease it by 0.05.