Safe Reinforcement Learning in Constrained Markov Decision Processes

Akifumi Wachi akifumi.wachi@ibm.com
IBM Research AI
Tokyo, Japan

Yanan Sui ysui@tsinghua.edu.cn
Tsinghua Univesity
Beijing, China

Abstract

Safe reinforcement learning has been a promising approach for optimizing the policy of an agent that operates in safety-critical applications. In this paper, we propose an algorithm, SNO-MDP, that explores and optimizes Markov decision processes under unknown safety constraints. Specifically, we take a stepwise approach for optimizing safety and cumulative reward. In our method, the agent first learns safety constraints by expanding the safe region, and then optimizes the cumulative reward in the certified safe region. We provide theoretical guarantees on both the satisfaction of the safety constraint and the near-optimality of the cumulative reward under proper regularity assumptions. In our experiments, we demonstrate the effectiveness of SNO-MDP through two experiments: one uses a synthetic data in a new, openly-available environment named GP-Safety-Gym, and the other simulates Mars surface exploration by using real observation data.[1]

Keywords: Reinforcement Learning, Markov Decision Process

1. Introduction

In many real applications, environmental hazards are first detected *in situ*. For example, a planetary rover exploring Mars does not obtain the high-resolution images at the time of the launch. In usual cases, after landing on Mars, the rover takes close-up images or observes terrain data. Leveraging the acquired data, ground operators identify whether each position is safe. Hence, for the fully automated operation, an agent must autonomously *explore* the environment and *guarantee* safety. However, in most cases, guaranteeing safety (i.e., surviving) is *not* the primary objective. The optimal policy for ensuring safety is often extremely conservative (e.g., stay at the current position). Even though avoiding hazards is an essential requirement, the primary objective is nonetheless to obtain rewards.

As a framework to solve this problem, safe reinforcement learning (safe RL, Garcıa and Fernández (2015)) has recently been noticed by the research community. The objective of safe RL is to maximize the cumulative reward while guaranteeing or encouraging safety. Especially in the problem settings in which the reward and safety functions are *unknown a priori*, however, a great deal of previous work (e.g., Wachi et al. (2018)) theoretically guarantees the satisfaction of the safety constraint, but the acquired policy is not necessarily

1. This paper is a short version of Wachi and Sui (2020), which will be presented in ICML 2020.

near-optimal in terms of the cumulative reward. In this paper, we propose a safe RL algorithm that guarantees a near-optimal cumulative reward while guaranteeing the satisfaction of the safety constraint as well.

Related work. As the research community tries to apply RL algorithms to real-world systems, safety issues have been highlighted. RL algorithms inherently require an agent to explore unknown state-action pairs, and algorithms that are agnostic with respect to safety may execute unsafe actions without deliberateness. Hence, it is important to develop algorithms that guarantee safety even during training, at least with high probability. A notable approach is *safe model-based RL* (Berkenkamp et al., 2017; Fisac et al., 2018). In this domain, safety is associated with a state constraint; thus, the resulting algorithm is well suited for such contexts as a drone learning how to hover. On the other hand, safe *model-free* RL has also been successful, especially in continuous control tasks. For example, Achiam et al. (2017) proposed the constrained policy optimization (CPO) algorithm while guaranteeing safety in terms of constraint satisfaction. Finally, several previous studies have addressed how to explore a safe space in an environment that is unknown a priori (Sui et al., 2015; Turchetta et al., 2016). This type of problem setting is well-suited for cases such as a robot exploring an uncertain environment (e.g., a planetary surface).

Our contributions. We propose a safe near-optimal MDP, SNO-MDP algorithm, for achieving a near-optimal cumulative reward while guaranteeing safety in a constrained MDP. This algorithm first explores the safety function and then optimizes the cumulative reward in the certified safe region. We further propose an algorithm called Early Stopping of Exploration of Safety (ES^2) to achieve faster convergence while maintaining probabilistic guarantees with respect to both safety and reward. We examine SNO-MDP by applying PAC-MDP analysis and prove that, with high probability, the acquired policy is near-optimal with respect to the cumulative reward while guaranteeing safety. We build an openly-available test-bed called GP-Safety-Gym for synthetic experiments. The safety and efficiency of SNO-MDP were evaluated with our experiments.

2. Problem Statement

A safety constrained MDP is defined as a tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, f, r, g, \gamma \rangle$, where $\mathcal{S}$ is a finite set of states $\{\boldsymbol{s}\}$, $\mathcal{A}$ is a finite set of actions $\{a\}$, $f : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is a deterministic state transition function, $r : \mathcal{S} \to (0, R_{\max}]$ is a bounded reward function, $g : \mathcal{S} \to \mathbb{R}$ is a safety function, and $\gamma \in \mathbb{R}$ is a discount factor. We assume that both the reward function r and the safety function g are *not known a priori*. At every time step $t \in \mathbb{N}$, the agent must be in a "safe" state. More concretely, for a state $\boldsymbol{s}_t$, the safety function value $g(\boldsymbol{s}_t)$ must be above a threshold $h \in \mathbb{R}$; that is, the safety constraint is represented as $g(\boldsymbol{s}_t) \geq h$.

A policy $\pi : \mathcal{S} \to \mathcal{A}$ maps a state to an action. The value of a policy is evaluated based on the discounted cumulative reward under the safety constraint. Let $V_{\mathcal{M}}$ denote the value function in the MDP, $\mathcal{M}$. In summary, we represent our problem as follows:

$$\max \quad V^{\pi}_{\mathcal{M}}(\boldsymbol{s}_t) = \mathbb{E}\left[\sum_{\tau=0}^{\infty} \gamma^{\tau} r(\boldsymbol{s}_{t+\tau}) \;\middle|\; \boldsymbol{s}_t \right] \qquad \text{s.t.} \quad g(\boldsymbol{s}_{t+\tau}) \geq h, \quad \forall \tau = [0, \infty].$$

Difficulties. In conventional safety-constrained RL algorithms, the safety function is assumed to be known a priori. The key difference lies in the fact that we need to explore a safety function that is unknown a priori while guaranteeing satisfaction of the safety constraint. However, it is intractable to solve the above problem without further assumptions. First of all, without prior information on the state-and-action pairs known to be safe, an agent cannot take any viable action at the very beginning. Second, if the safety function does not exhibit any regularity, then the agent cannot infer the safety of decisions.

Assumptions. To overcome the difficulties mentioned above, we adopt two assumptions from Sui et al. (2015) and Turchetta et al. (2016). For the first difficulty, we simply assume that the agent starts in an initial set of states, S_0, that is known a priori to be safe. Second, we assume regularity for the safety function. Formally speaking, we assume that the state space $\mathcal{S}$ is endowed with a positive definite kernel function, k^g, and that the safety function has a bounded norm in the associated reproducing kernel Hilbert space (RKHS, Schölkopf and Smola (2001)). The kernel function, k^g is employed to capture the regularity of the safety function. Finally, we further assume that the safety function g is L-Lipschitz continuous with respect to some distance metric $d(\cdot, \cdot)$ on $\mathcal{S}$. As with the safety function, we also assume that the reward function has a bounded norm in the associated RKHS, and that its regularity is captured by another positive definite kernel function, k^r.

The above assumptions allow us to characterize the reward and safety functions by using Gaussian processes (GPs, see Rasmussen (2004)). By using the GP models, the values of r and g at unobserved states are predicted according to previously observed functions' values. An advantage of leveraging GPs is that we can obtain both optimistic and pessimistic measurements of the two functions by using the inferred means and variances. A GP is specified by its mean, $\mu(\boldsymbol{s})$, and covariance, $k(\boldsymbol{s}, \boldsymbol{s}')$. The reward and safety functions are thus modeled as $r(\boldsymbol{s}) = \mathcal{GP}(\mu^r(\boldsymbol{s}), k^r(\boldsymbol{s}, \boldsymbol{s}'))$ and $g(\boldsymbol{s}) = \mathcal{GP}(\mu^g(\boldsymbol{s}), k^g(\boldsymbol{s}, \boldsymbol{s}'))$. Without loss of generality, let $\mu(\boldsymbol{s}) = 0$ for all $\boldsymbol{s} \in \mathcal{S}$. For the reward and safety functions, we respectively model the observation noise as $y^r = r(\boldsymbol{s}) + n^r$ and $y^g = g(\boldsymbol{s}) + n^g$, where $n^r \sim \mathcal{N}(0, \sigma_r^2)$ and $n^g \sim \mathcal{N}(0, \sigma_g^2)$. The posteriors over r and g are computed on the basis of t observations at states $\{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_t\}$. Then, for both the reward and safety functions, the posterior mean, variance, and covariance are respectively represented as $\boldsymbol{\mu}_t(\boldsymbol{s}) = \boldsymbol{k}_t^\top(\boldsymbol{s})(\boldsymbol{K}_t + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{y}_t$, $\boldsymbol{\sigma}_t(\boldsymbol{s}) = \boldsymbol{k}_t(\boldsymbol{s}, \boldsymbol{s})$, $\boldsymbol{k}_t(\boldsymbol{s}, \boldsymbol{s}') = \boldsymbol{k}(\boldsymbol{s}, \boldsymbol{s}') - \boldsymbol{k}_t^\top(\boldsymbol{s})(\boldsymbol{K}_t + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{k}_t(\boldsymbol{s}')$, where $\boldsymbol{k}_t(\boldsymbol{s}) = [k(\boldsymbol{s}_1, \boldsymbol{s}), \ldots, k(\boldsymbol{s}_t, \boldsymbol{s})]^\top$, and $\boldsymbol{K}_t$ is the positive definite kernel matrix.

3. Background

We define two kinds of predicted safe spaces inferred by a GP as in Turchetta et al. (2019). First, we consider a *pessimistic* safe space, which contains states identified as safe with a greater probability than a pre-defined confidence level. Second, we derive an *optimistic* safe space that includes all states that may be safe with even a small probability.

Predicted pessimistic safe space. We use the notion of a safe space in Turchetta et al. (2016) as a *predicted pessimistic safe space*. For the probabilistic safety guarantee, two sets are defined. The first set, S_t^-, simply contains the states that satisfy the safety constraint with high probability. The second one, $\mathcal{X}_t^-$, additionally considers the ability to reach states in S_t^- (i.e., reachability) and the ability to return to the previously identified safe set, $\mathcal{X}_{t-1}^-$

(i.e., returnability). The algorithm probabilistically guarantees safety by allowing the agent to visit only states in $\mathcal{X}_t^-$.

Predicted optimistic safe space. As defined in Wachi et al. (2018) and Turchetta et al. (2019), an *optimistic* safe space has rich information for inferring the safety function. Let $\mathcal{X}_t^+$ denote the predicted optimistic safe space. Intuitively, $\mathcal{X}_t^+$ contains all states that may turn out to be safe even if the probability is low. In other words, $\mathcal{S} \setminus \mathcal{X}_t^+$ contains states that are unsafe with high probability.

4. Algorithm

We now introduce our proposed algorithm, SNO-MDP, for achieving a near-optimal policy while guaranteeing safety. We first give an overview of SNO-MDP. We extend a stepwise approach in Sui et al. (2018) from state-less to stateful settings. Basically, our algorithm consists of two steps. In the first step, the agent expands the pessimistic safe region while guaranteeing safety). Next, it explores and exploits the reward in the safe region certified in the first step. The reason for this stepwise approach is that we can neglect uncertainty related to the a priori unknown safety function once the safe region is fixed. However, a pure stepwise approach does not stop exploring the safe region until the convergence of the GP confidence interval). This formulation often requires the agent to execute a great number of actions for exploring safety. Hence, to achieve near-optimality while executing a smaller number of actions, we also propose the ES^2 algorithm.

4.1 Exploration of Safety (Step 1)

First, we consider how to explore the safety function. As a scheme to expand the safe region, we consider "expanders" as in Sui et al. (2015) and Turchetta et al. (2016). Expanders are states that may expand the predicted safe region, which is defined as $G_t = \{s \in \mathcal{X}_t^- \mid e_t(s) > 0\}$, where $e_t(s) = |s' \in \mathcal{S} \setminus S_t^- \mid u_t(s) - Ld(s, s') \geq h|$.

The *efficiency* of expanding the safe region is measured by the width of the safety function's confidence interval, defined as $w_t(\boldsymbol{s}) = u_t(s) - l_t(s)$. The agent safely and efficiently expands the safe region by sampling the state with the maximum value of w among the expanders, G_t. Hence, the agent sets the temporal goal according to $\xi = \arg\max_{\boldsymbol{s} \in G_t} w_t(\boldsymbol{s})$. Then, within the predicted safe space $\mathcal{X}_t^-$, it chooses a path to get to ξ from the current state $\boldsymbol{s}_{t-1}$ so as to minimize the cost (e.g., the path length). In our experiment, we simply minimize the path length. By defining the cost as related to w (e.g., $1/w$), however, the agent could explore safety more actively on the way to ξ.

The previous work Sui et al. (2015); Turchetta et al. (2016); Sui et al. (2018) terminated safety exploration when the desired accuracy was achieved for every state in G_t; that is,

$$\max_{s \in G_t} w_t(\boldsymbol{s}) \leq \epsilon_g. \tag{1}$$

Unfortunately, this termination condition often requires a great number of iterations. For the purpose of maximizing the cumulative reward, it often leads to the loss of reward. Therefore, in Section 4.3, we propose the ES^2 algorithm to improve this point.

4.2 Exploration and Exploitation of Reward (Step 2)

Once expansion of the safe region is completed, the agent guarantees safety as long as it is in $\mathcal{X}^-$ and does not have to expand the safe region anymore. Hence, all we have to do is optimize the cumulative reward in $\mathcal{X}^-$. As such, a simple approach is to follow the *optimism in the face of uncertainty* principle as in Strehl and Littman (2008) and Auer and Ortner (2007), then to consider the "exploration bonus" represented by R-MAX (Brafman and Tennenholtz, 2002) and Bayesian Exploration Bonus (BEB, Kolter and Ng (2009)).

Specifically, we optimize the policy by *optimistically* measuring the reward with the (probabilistic) upper confidence bound, $U_t(\boldsymbol{s}) := \mu_t^r(\boldsymbol{s}) + \alpha_{t+1}^{1/2} \cdot \sigma_t^r(\boldsymbol{s})$. In this reward setting, the second term on the right-hand side corresponds to the exploration bonus. For balancing the exploration and exploitation in terms of reward, we solve the following Bellman equation:

$$J_{\mathcal{X}}^*(\boldsymbol{s}_t, \boldsymbol{b}_t^r, \boldsymbol{b}_t^g) = \max_{s_{t+1} \in \mathcal{X}_{t^*}^-} \big[U_t(\boldsymbol{s}_{t+1}) + \gamma J_{\mathcal{X}}^*(\boldsymbol{s}_{t+1}, \boldsymbol{b}_t^r, \boldsymbol{b}_t^g) \big],$$

where $\boldsymbol{b}^r = (\mu^r, \sigma^r)$ and $\boldsymbol{b}^g = (\mu^g, \sigma^g)$ are the beliefs over reward and safety, respectively. Also, t^* is the time step when the termination condition (1) is satisfied. Note that $\boldsymbol{b}^r$ and $\boldsymbol{b}^g$ are not updated; hence, we can solve the above equation with standard algorithms.

4.3 Early Stopping of Exploration of Safety (ES^2)

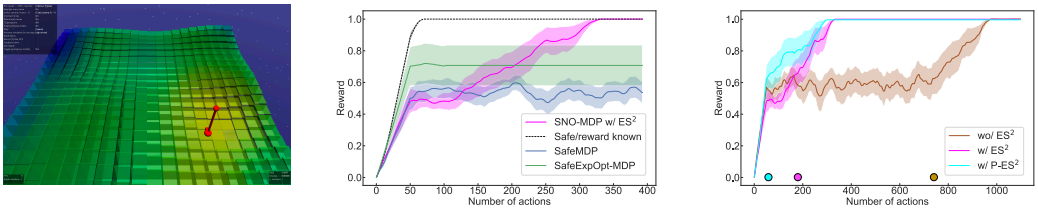
The existing safe exploration algorithms (Sui et al., 2015; Turchetta et al., 2016) continue exploring the state space until convergence of the confidence interval, w, which generally leads to a large number of iterations. Our primary objective is to maximize the cumulative reward; hence, we should stop exploring safety if further exploration will not lead to maximizing the cumulative reward.

While exploring the safety function, we check whether migration of the step can be conducted. As such, we consider the following additional MDP, $\mathcal{M}_y = \langle \mathcal{X}^+, \mathcal{A}, f, r', g, \gamma \rangle$. The differences from the original MDP, $\mathcal{M}$, lie in the state space and the reward function. The state space of $\mathcal{M}_y$ is defined as the optimistic safe space (i.e., $\mathcal{X}^+$), while the reward function is defined as follows:

$$r' := \begin{cases} \mu^r + \alpha^{1/2}\sigma^r & \text{if } \boldsymbol{s} \in \mathcal{X}_t^+ \setminus \mathcal{X}_t^-, \\ \mu^r - \alpha^{1/2}\sigma^r & \text{if } \boldsymbol{s} \in \mathcal{X}_t^-. \end{cases} \tag{2}$$

This definition of the reward function encourages the agent to explore outside the predicted safe space, $\mathcal{X}_t^-$. Using the new MDP above, we consider the set of states that the agent will visit at the next time step, defined as $\mathcal{Y}_t = \{\boldsymbol{s}' \in \mathcal{S}^+ \mid \forall \boldsymbol{s} \in \mathcal{X}_t^- : \boldsymbol{s}' = f(\boldsymbol{s}, \pi_y^*(a \mid \boldsymbol{s}))\}$, where π_y^* is the optimal policy for $\mathcal{M}_y$, obtained by maximizing the $V_{\mathcal{M}_y}(\boldsymbol{s}_t) = \max_{s_{t+1} \in \mathcal{X}_t^+} [\ r'(\boldsymbol{s}_{t+1}) + \gamma V_{\mathcal{M}_y}(\boldsymbol{s}_{t+1})\ ]$. Finally, we stop exploring the safety function if $\mathcal{Y}_t \subseteq \mathcal{X}_t^-$ holds. Intuitively, we stop expanding the safe space if the direction of the optimal policy for $\mathcal{M}_y$ heads for the inside of $\mathcal{X}_t^-$. If the agent tries to stay in $\mathcal{X}_t^-$ even under the condition that the reward is defined as in (2), then we do not have to expand the safe region anymore. When the ES^2 algorithm confirms satisfaction of the above condition, we move on to the next step and then optimize the cumulative reward in $\mathcal{Y}_t$; that is, $J_{\mathcal{Y}}^*(\boldsymbol{s}_t, \boldsymbol{b}_t^r, \boldsymbol{b}_t^g) = \max_{s_{t+1} \in \mathcal{Y}_t} \big[U_t(\boldsymbol{s}_{t+1}) + \gamma J_{\mathcal{Y}}^*(\boldsymbol{s}_{t+1}, \boldsymbol{b}_t^r, \boldsymbol{b}_t^g) \big]$.

We also developed P-ES^2 algorithm, which empirically works better than ES^2 algorithm by modeling a probability of a state being safe.



(a) GP-Safety-Gym. (b) Performance comparison. (c) Effects of ES^2 and P-ES^2.

Figure 1: Experiment with synthetic data. (a) An example screen capture from the GP-Safety-Gym environment. (b) Average reward over the episodes, comparing the performance of SNO-MDP with ES^2 and the other baselines. (c) Average reward over the episodes, showing the effects of ES^2 and P-ES^2. The colored circles represent when the transition from safe exploration to reward optimization happens for each method.

5. Experiment

In this section, we evaluate the performance of SNO-MDP in two experiments.

Settings. We constructed a new open-source environment for safe RL simulations named GP-Safety-Gym. This environment was built based on OpenAI Safety-Gym Ray et al. (2019). As shown in Figure 1(a), GP-Safety-Gym represents the reward by a color (yellow: high; green: medium; blue: low), and the safety by height. We considered a 20×20 square grid in which the reward and safety functions were randomly generated. At every time step, an agent chose an action from *stay*, *up*, *right*, *down*, and *left*. The agent predicted the reward and safety functions by using different kernels on the basis of previous observations. In this simulation, we allowed the agent to observe the reward and safety function values of the current state and neighboring states.

Results. Figure 1(b) compares the performance of SNO-MDP and the baselines in terms of the reward. SNO-MDP achieved the optimal reward after shifting to the stage of reward optimization, which outperforms SafeMDP (Turchetta et al. (2016)) and SafeExpOpt-MDP (Wachi et al. (2018)) in terms of reward after sufficiently large number of time steps. The SafeMDP agent did not aim to maximize the cumulative reward, and the SafeExpOpt-MDP agent was sometimes stucked in a local optimum when the expansion of the safe region was insufficient. Figure 1(c) shows the empirical performance of the ES^2 algorithm. Also, all methods, including the baselines, did *not* take any unsafe actions.

6. Conclusion

We have proposed SNO-MDP, a stepwise approach for exploring and optimizing a safety-constrained MDP. Theoretically, we proved a bound of the sample complexity to achieve ϵ_V-closeness to the optimal policy while guaranteeing safety, with high probability. We also proposed the ES^2 algorithm for improving the efficiency in obtaining rewards. We developed an open-source environment, GP-Safety-Gym, to test the effectiveness of SNO-MDP . We also demonstrated the advantages of SNO-MDP using the real Mars terrain data.

References

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International Conference on Machine Learning (ICML)*, 2017.

Peter Auer and Ronald Ortner. Logarithmic online regret bounds for undiscounted reinforcement learning. In *Neural Information Processing Systems (NeurIPS)*, 2007.

Felix Berkenkamp, Matteo Turchetta, Angela Schoellig, and Andreas Krause. Safe model-based reinforcement learning with stability guarantees. In *Neural Information Processing Systems (NeurIPS)*, 2017.

Ronen I Brafman and Moshe Tennenholtz. R-max - a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research (JMLR)*, pages 213–231, 2002.

Jaime F Fisac, Anayo K Akametalu, Melanie N Zeilinger, Shahab Kaynama, Jeremy Gillula, and Claire J Tomlin. A general safety framework for learning-based control in uncertain robotic systems. *IEEE Transactions on Automatic Control*, 2018.

Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research (JMLR)*, 16(1):1437–1480, 2015.

J Zico Kolter and Andrew Y Ng. Near-Bayesian exploration in polynomial time. In *International Conference on Machine Learning (ICML)*, 2009.

Carl Edward Rasmussen. Gaussian processes in machine learning. In *Advanced Lectures on Machine Learning*, pages 63–71. Springer, 2004.

Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking safe exploration in deep reinforcement learning. 2019.

Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

Alexander L Strehl and Michael L Littman. An analysis of model-based interval estimation for Markov decision processes. *Journal of Computer and System Sciences*, 74(8):1309–1331, 2008.

Yanan Sui, Alkis Gotovos, Joel W Burdick, and Andreas Krause. Safe exploration for optimization with Gaussian processes. In *International Conference on Machine Learning (ICML)*, 2015.

Yanan Sui, Vincent Zhuang, Joel W. Burdick, and Yisong Yue. Stagewise safe Bayesian optimization with Gaussian processes. In *International Conference on Machine Learning (ICML)*, 2018.

Matteo Turchetta, Felix Berkenkamp, and Andreas Krause. Safe exploration in finite Markov decision processes with Gaussian processes. In *Neural Information Processing Systems (NeurIPS)*, 2016.

Matteo Turchetta, Felix Berkenkamp, and Andreas Krause. Safe exploration for interactive machine learning. In *Neural Information Processing Systems (NeurIPS)*, 2019.

Akifumi Wachi and Yanan Sui. Safe reinforcement learning in constrained Markov decision processes. In *International Conference on Machine Learning (ICML)*, 2020.

Akifumi Wachi, Yanan Sui, Yisong Yue, and Masahiro Ono. Safe exploration and optimization of constrained MDPs using Gaussian processes. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2018.