Robustness Verification of Deep Reinforcement Learning Based Control Systems Using Reward Martingales

Dapeng Zhi[1], Peixin Wang[2*] , Cheng Chen[1], Min Zhang[1*]

[1]Shanghai Key Laboratory of Trustworthy Computing, East China Normal University
[2]University of Oxford
zhi.dapeng@163.com, peixin.wang@cs.ox.ac.uk, {chchen,zhangmin}@sei.ecnu.edu.cn

Abstract

Deep Reinforcement Learning (DRL) has gained prominence as an effective approach for control systems. However, its practical deployment is impeded by state perturbations that can severely impact system performance. Addressing this critical challenge requires robustness verification about system performance, which involves tackling two quantitative questions: (i) how to establish guaranteed bounds for expected cumulative rewards, and (ii) how to determine tail bounds for cumulative rewards. In this work, we present the first approach for robustness verification of DRL-based control systems by introducing *reward martingales*, which offer a rigorous mathematical foundation to characterize the impact of state perturbations on system performance in terms of cumulative rewards. Our verified results provide provably quantitative certificates for the two questions. We then show that reward martingales can be implemented and trained via neural networks, against different types of control policies. Experimental results demonstrate that our certified bounds tightly enclose simulation outcomes on various DRL-based control systems, indicating the effectiveness and generality of the proposed approach.

Introduction

Deep Reinforcement Learning (DRL) is gaining widespread adoption in various control systems, including safety-critical ones like power systems (Zhang, Tu, and Liu 2023; Wan et al. 2023) and traffic signal controllers (Liu and Ding 2022; Chen et al. 2020). As these systems collect state information via sensors, uncertainties inevitably originate from sensor errors, equipment inaccuracy, or even adversarial attacks (Zhang et al. 2020; Wan, Zeng, and Sun 2022; Zhang et al. 2023). In real-world scenarios, the robustness guarantee of their performance is of utmost importance when they are subjected to reasonable environmental perturbations and adversarial attacks. Failing to do so could lead to critical errors and a significant decline in performance, which may cause fatal consequences in safety-critical applications.

A DRL-based control system's robustness can be reflected in its performance variation, i.e., the cumulative rewards, when the system is perturbed (Lütjens, Everett, and How 2019). The robustness verification refers to two quantitative questions: (i) how to establish guaranteed bounds for expected cumulative rewards, and (ii) how to determine their tail bounds. However, the verification is a very challenging task. Firstly, DRL-based control systems are complex cyber-physical systems, making formal verification difficult (Deshmukh and Sankaranarayanan 2019). Secondly, the inclusion of opaque AI models like Deep Neural Networks (DNNs) adds complexity to the problem (Larsen et al. 2022). Thirdly, performance is measured statistically rather than by analytical calculations, lacking theoretical guarantees.

In this work, we propose a novel approach for formally verifying the robustness of DRL-based control systems. By leveraging the concept of *martingales* from probabilistic programming (Chakarov and Sankaranarayanan 2013; Wang et al. 2019; Wang 2022), we establish provable upper and lower bounds for the expected cumulative rewards of DRL-based control systems under state perturbations. Specifically, we define *upper reward supermartingales (URS)* and *lower reward submartingales (LRS)* and prove they provide theoretical guarantees in the system's certified reward range. We then extend our analysis to encompass tail bounds of rewards, utilizing a combination of martingales and Hoeffding's inequality (Hoeffding 1994). This refined approach can derive upper bounds for tail probabilities that show the deviation of system performance from predefined thresholds, offering a comprehensive understanding of the system's robustness.

We further show that reward martingales can be efficiently implemented and trained as DNNs, as ranking martingales are trained (Lechner et al. 2022), against various control policies. Given a DRL-based control system, we define a corresponding loss function and train a DNN repeatedly until the DNN satisfies the conditions of being a reward martingale or timeout. We identify that computing expected rewards is the difficult part in checking whether a trained DNN is a reward martingale and it varies in the training approaches of policies. If policies are implemented by DNNs on infinite and continuous state space, we take advantage of the over-approximation-based method (Lechner et al. 2022). We also propose an analytical method to compute expected values precisely when policies are trained on discretized abstract state space in recently emerging approaches (Jin et al. 2022; Li et al. 2022; Drews, Albarghouthi, and D'Antoni 2020).

We intensively evaluate the effectiveness of our approach on four classic control problems, namely, MountainCar, Cart-

*Corresponding Author.

Pole, B1, and B2. Through rigorous quantitative robustness verification using our proposed method, we assess the performance of the corresponding DRL-based control systems. To demonstrate our approach's effectiveness, we compare the verified lower and upper bounds, and tail bounds with the performance achieved through simulations under the same settings. Our experimental results demonstrate that the verified bounds tightly enclose the simulation outcomes.

In summary, this work makes three major contributions:

- We introduce reward martingales and prove that they analytically characterize both reward bounds and tail bounds to the performance of perturbed DRL-based control systems, rendering us the first robustness verification approach to those systems.
- We show that reward martingales can be represented and efficiently trained in the form of deep neural networks and propose corresponding validation approaches for policies trained by two different approaches.
- We intensively evaluate our approach on four classic control problems with control policies under two different training approaches, demonstrating the effectiveness and generality of the proposed approach.

Related Work

Qualitative Verification of DRL-based Control Systems Formal verification of DRL-based control systems has received increasing attention for safety assurance in recent years. Jin et al. (2022) proposed a CEGAR-driven training and verification framework that guarantees that the trained systems satisfy the properties predefined in ACTL formulas. Bacci (2022) developed formal models of controllers executing under uncertainty and proposed new verification techniques based on abstract interpretation. Corsi, Marchesini, and Farinelli (2021) provided a new formulation for the safety properties to ensure that an agent always makes rational decisions. They focus on qualitative verification for specific properties but lack quantitative guarantees.

Robust Training of DRL Systems Several attempts are made to improve DRL systems' robustness by means of formal verification (Oikarinen et al. 2021; Kumar, Levine, and Feizi 2022). For instance, Lütjens, Everett, and How (2019) proposed to compute guaranteed lower bounds on state-action values to determine the optimal action under a worst-case deviation in input space. Oikarinen et al. (2021) designed adversarial loss functions by leveraging existing formal verification bounds w.r.t. neural network robustness. Zhang et al. (2020) studied the fundamental properties of state-adversarial Markov decision processes and developed a theoretically principled policy regularization. These approaches focused on robust training rather than verification, and they have to rely on simulation to demonstrate the effectiveness of their approaches in robustness improvement.

Quantitative Verification in Stochastic Control Systems via Martingales Robustness verification of control systems is essentially a quantitative verification problem, which provides certified guarantees to systems' quantitative properties such as stabilization time. Some studies emerged in this direction. Lechner et al. (2022) considered the problem of formally verifying almost-sure (a.s.) asymptotic stability in discrete-time nonlinear stochastic control systems and presented an approach for general nonlinear stochastic control problems with two aspects: using ranking supermartingales (RSMs) to certify a.s. asymptotic stability and presenting a method for learning neural network RSMs. Zikelic et al. (2023) studied the problem of learning controllers for discrete-time nonlinear stochastic dynamical systems with formal reach-avoid guarantees by combining and generalizing stability and safety guarantees with a tolerable probability threshold over the infinite time horizon in general Lipschitz continuous systems. However, these works do not consider system robustness, a non-trivial property of DRL-based control systems.

Preliminaries

DRL-Based Control Systems We consider DRL-based control systems where the control policies are implemented by neural networks (NNs) and suppose the networks are trained. Formally, a DRL-based control system is a tuple $M = (S, S_0, S_g, A, \pi, f, R)$, where $S \subseteq \mathbb{R}^n$ is the set of system states (possibly infinite), $S_0 \subseteq S$ (resp. $S_g \subseteq S$) is the set of initial (resp. terminal) states and $S_0 \cap S_g = \emptyset$, A is the set of actions, $\pi : S \to \mathbb{R}$ is the trained policy implemented by a neural network, $f : S \times A \to S$ is the system dynamics, and $R : S \times A \times S \to \mathbb{R}$ is the reward function. [1]

A trained DRL-based control system M is a decision-making system that continuously interacts with the environment. At each time step $t \in \mathbb{N}_0$, it observes a state s_t and feeds s_t into its planted NN to compute the optimal action $a_t = \pi(s_t)$ that shall be taken. Action a_t is then performed, which transits s_t into the next state $s_{t+1} = f(s_t, a_t)$ via the system dynamics f and earns some reward $r_{t+1} = R(s_t, a_t, s_{t+1})$. Given an initial state $s_0 \in S_0$, a sequence of states generated during interaction is called an *episode*, denoted as $e = \{s_t\}_{t \in \mathbb{N}_0}$.

State Perturbations During interaction with environments, the observed states of systems may be perturbed and actions are computed based on the perturbed states. Formally, an observed state at time t is $\hat{s}_t := s_t + \delta_t$ where $\delta_t \sim \mu$ and μ is a probability distribution over $\mathbb{R}^n$. Due to perturbation, the actual successor state is $s_{t+1} := f(s_t, \hat{a}_t)$ with $\hat{a}_t := \pi(\hat{s}_t)$ and the reward is $r_{t+1} := R(s_t, \hat{a}_t, s_{t+1})$. Note that the successor state and reward are calculated according to the actual state and the action on the perturbed state, and this update is common (Zhang et al. 2020). We then denote a DRL-based control system M perturbed by the noise distribution μ as $M_\mu = (S, S_0, S_g, A, \pi, f, R, \mu)$.

Probability Space Given an M_μ, for each $s_0 \in S_0$, there exists a *probability space* $(\Omega_{s_0}, \mathcal{F}_{s_0}, \mathbb{P}_{s_0})$ such that Ω_{s_0} is the set of all episodes that start from s_0 by the environmental interaction, $\mathcal{F}_{s_0}$ is a σ-algebra over Ω_{s_0} (i.e., a collection of subsets of Ω_{s_0} that contains the empty set $\emptyset$ and is closed under complementation and countable union), and $\mathbb{P}_{s_0} : \mathcal{F} \to$

[1] Here we focus on deterministic system dynamics and policies, and leave the analysis of probabilistic ones as future work.

$[0, 1]$ is a probability measure on $\mathcal{F}_{s_0}$. We also denote the expectation operator in this probability space by $\mathbb{E}_{s_0}$.

Termination Time When states are perturbed, actions may become sub-optimal, which may cause non-termination or premature termination. Thus, prerequisites for studying the robustness of DRL-based control systems are to guarantee the system is terminating and know its *termination time*. Intuitively, the termination time of an episode is the number of steps it takes for the episode to reach the terminal set or ∞ if it never reaches S_g.

Formally, the *termination time* of an M_μ is a random variable defined on episodes as $T(\{s_t\}_{t\in\mathbb{N}_0}) := \min\{t \in \mathbb{N}_0 \mid s_t \in S_g\}$. We define $\min \emptyset = \infty$. A control system is *finitely terminating* if it has finite expected termination time over all episodes, i.e., $\mathbb{E}_{s_0}[T] < \infty$ for all states $s_0 \in S_0$. Besides, a system has the *concentration property* if there exist two constants $a, b > 0$ such that for sufficiently large $n \in \mathbb{N}$, we have $\mathbb{P}_{s_0}(T > n) \leq a \cdot \exp(-b \cdot n)$ for all states $s_0 \in S_0$, i.e. if the probability that the system executes n steps or more decreases exponentially as n grows.

Problem Formulation

Model Assumptions Given a DRL-based control system, it is assumed that its state space S is compact in the Euclidean topology of $\mathbb{R}^n$, its system dynamics f and trained policy π are Lipschitz continuous. This assumption is common in control theory (Zikelic et al. 2023). Besides, we further assume that once the system state enters S_g, it will stop and no more actions will be taken, i.e., for any $s_T \in S_g$, $s_{T+1} = s_T$. The control systems of interest are assumed to be finitely terminating, which can be checked by the stability verification approach (Lechner et al. 2022). For perturbation, we assume that a noise distribution μ either has bounded support or is a product of independent univariate distributions.

Definition 1 (Cumulative Rewards)**.** *Given an M_μ with termination time T, its* cumulative reward *is a random variable defined on episodes as $\mathcal{R}(e) := \sum_{t=0}^{T} r_t$, where $e = \{s_t\}_{t=0}^{T}$ is an episode and r_t is the step-wise reward of e that is determined by the reward function $R(\cdot)$ in M_μ with $r_0 \equiv 0$.*

Intuitively, the cumulative reward is the sum of all step-wise rewards until the system reaches a terminal state. It is a random variable and varies from different episodes.

Robustness Verification Problems of M_μ Given an M_μ and an initial state $s_0 \in S_0$, we are interested in the following two robustness problems:

1. What are the upper and lower bounds for $\mathbb{E}_{s_0}[\mathcal{R}]$?
2. Given a reward c, what is the tail bound of $\mathbb{P}_{s_0}(\mathcal{R} \geq c)$ (resp. $\mathbb{P}_{s_0}(\mathcal{R} \leq c)$) if c is larger (resp. smaller) than the upper (resp. lower) bound of $\mathbb{E}_{s_0}[\mathcal{R}]$?

The first problem concerns certified upper and lower bounds of expected cumulative rewards when systems are perturbed. The second problem considers two cases. Provided a cumulative reward c that is greater than the upper bound of the expected cumulative reward $\mathbb{E}_{s_0}[\mathcal{R}]$, we are interested in the tail probability that a system can achieve a reward greater than c. The dual problem is to compute the tail probability that the system can achieve a cumulative reward lower than c, when c is less than the lower bound of $\mathbb{E}_{s_0}[\mathcal{R}]$. A higher tail probability implies worse robustness because it indicates a higher probability that the reward gets out of the certified range of expected cumulative rewards.

Reward Martingales and Fundamentals

In this section, we present our theoretical results about the two robustness verification problems by introducing the notion of *reward martingales*. It is the foundation of reducing the robustness verification problems of perturbed DRL-based control systems to the analysis of a stochastic process.

In the following, we fix a perturbed DRL-based control system $M_\mu = (S, S_0, S_g, A, \pi, f, R, \mu)$ and denote the difference $S \setminus S_g$ by $\overline{S_g}$ for the set of non-terminal states in S.

To define reward martingales, we first need the notion of pre-expectation of functions. Given a function $h(\cdot)$, the pre-expectation $pre_h(\cdot)$ of $h(\cdot)$ is the reward of the current step plus the expected value of $h(\cdot)$ in the next step of the system.

Definition 2 (Pre-Expectation)**.** *Given an M_μ and a function $h : S \to \mathbb{R}$, the pre-expectation of h is a function $pre_h : S \to \mathbb{R}$, such that:*

$$pre_h(s) = \begin{cases} h(s) & \text{if } s \in S_g \\ r + \mathbb{E}_{\delta\sim\mu}[h(f(s, \pi(s+\delta)))] & \text{if } s \in \overline{S_g} \end{cases}$$

where, $r = R(s, \pi(s), f(s, \pi(s)))$ is the reward of performing action $\pi(s)$ in state s.

We next define the notion of reward martingales. First, we begin with the definition of URS which can be served as an upper bound for the expected cumulative reward of M_μ.

Definition 3 (Upper Reward Supermartingales, **URS**)**.** *Given an M_μ, a function $h : S \to \mathbb{R}$ is an upper reward supermartingale (URS) of M_μ if there exist $K, K' \in \mathbb{R}$ such that:*

$$\forall s \in S_g, K \leq h(s) \leq K' \qquad \text{(Boundedness)}$$
$$\forall s \in \overline{S_g}, pre_h(s) \leq h(s) \qquad \text{(Decreasing Pre-Expectation)}$$

Intuitively, the first condition says that the values of the URS at terminal states should always be bounded, and the second condition specifies that for all non-terminal states, the pre-expectation is no more than the value of the URS itself.

Similar to the definition of URS (Definition 3), we define LRS as follows and will employ it as a lower bound for the expected cumulative reward of M_μ.

Definition 4 (Lower Reward Submartingales, **LRS**)**.** *Given an M_μ, a function $\eta : S \to \mathbb{R}$ is a lower reward submartingale (LRS) of M_μ if there exist $K, K' \in \mathbb{R}$ such that:*

$$\forall s \in S_g . K \leq \eta(s) \leq K' \qquad \text{(Boundedness)}$$
$$\forall s \in \overline{S_g} . pre_\eta(s) \geq \eta(s) \qquad \text{(Increasing Pre-Expectation)}$$

Compared with the definition of URS, the only difference is that the second condition of LRS specifies that the pre-expectation is no less than the value of the LRS itself at all non-terminal states. We call K, K' the bounds of h (resp. η) if h (resp. η) is a URS (resp. LRS), correspondingly.

Definition 5 (Difference-boundedness)**.** *Given an M_μ and a function $h : S \to \mathbb{R}$, h is difference-bounded if there exists $m \in \mathbb{R}$ such that for any state $s \in S$, $|h(f(s, \pi(s))) - h(s)| \leq m$.*

Based on the URS and LRS, we have Theorem 1, stating that there must exist upper and lower bounds of the expected cumulative rewards of the perturbed system when we can calculate URS and LRS for the system.

Theorem 1 (Bounds for Expected Cumulative Rewards)**.** *Suppose an M_μ has a difference-bounded URS (resp. LRS) h (resp. η) and $K, K' \in \mathbb{R}$ are the bounds of h (resp. η). For each state $s_0 \in S_0$, we have*

$$\mathbb{E}_{s_0}[\mathcal{R}] \leq h(s_0) - K. \qquad \text{(Upper Bound)}$$

$$(\textit{resp. } \mathbb{E}_{s_0}[\mathcal{R}] \geq \eta(s_0) - K') \qquad \text{(Lower Bound)}$$

Proof Sketch. For upper bounds, we define the stochastic process $\{X_n\}_{n=0}^{\infty}$ as $X_n := h(\overline{s}_n)$, where h is an URS and $\overline{s}_n$ is a random (vector) variable representing value(s) of the state at the n-th step of an episode. Furthermore, we construct the stochastic process $\{Y_n\}_{n=0}^{\infty}$ such that $Y_n := X_n + \sum_{i=0}^{n} r_i$, where r_i is the reward of the i-th step. Let T be termination time of M_μ. We prove that $\{Y_n\}_{n=0}^{\infty}$ satisfies the prerequisites of the Optional Stopping Theorem (OST) (Williams 1991). This proof depends on the assumption that M_μ is finitely terminating and h is difference-bounded. Then by applying OST, we have that $\mathbb{E}[Y_T] \leq \mathbb{E}[Y_0]$. By the boundedness condition in Definition 3, we obtain that $\mathcal{R} = \sum_{i=0}^{T} r_i = Y_T - X_T \leq Y_T - K$. Finally, we conclude that $\mathbb{E}_{s_0}[\mathcal{R}] \leq \mathbb{E}[Y_T] - K \leq \mathbb{E}[X_0] - K = h(\overline{s}_0) - K$. The proof of lower bounds is similar. □

Theorem 2 (Tail Bounds for Cumulative Rewards)**.** *Suppose that an M_μ has the concentration property and a difference-bounded URS (*resp. *LRS) h (resp. η) with bounds $K, K' \in \mathbb{R}$. Given an initial state $s_0 \in S_0$, if a reward $c > h(s_0) - K$ (resp. $c < \eta(s_0) - K'$), we have*

$$\mathbb{P}_{s_0}(\mathcal{R} \geq c) \leq \alpha + \beta \cdot \exp(-\gamma \cdot c^2) \qquad (1)$$

$$(\text{resp. } \mathbb{P}_{s_0}(\mathcal{R} \leq c) \leq \alpha + \beta \cdot \exp(-\gamma \cdot c^2) \qquad (2)$$

where, α, β, γ are positive constants derived from M_μ, the concentration property and h (resp. η), respectively.

To prove Eq. (1), we construct a stochastic process $\{Y_n\}_{n=0}^{\infty}$ such that $Y_n := h(\overline{s}_n) + \sum_{i=0}^{n} r_i$ where h is an URS, $\overline{s}_n$ and r_i are defined as those in the proof sketch of Theorem 1. By Definition 3, we prove that $\{Y_n\}_{n=0}^{\infty}$ is a supermartingale. Then by the difference-bounded property of h (Definition 5), we derive the upper bound of $\mathbb{P}_{s_0}(\mathcal{R} \geq c)$ by the concentration property of M_μ and Hoeffding's Inequality on Martingales (Hoeffding 1994). Eq. (2) is obtained in the same manner.

Neural Network-Based Reward Martingales

Martingales are not necessarily polynomial functions and can be as complex as deep neural networks, as shown by the pioneering works (Abate, Giacobbe, and Roy 2021; Lechner et al. 2022; Zikelic et al. 2023; Dawson, Gao, and Fan 2023). Likewise, we show that reward martingales can be also achieved in the form of DNNs.

Our method consists of two modules that alternate within a loop: training and validating. In each loop iteration, we train a candidate reward martingale in the form of a neural network which is then passed to the validation. If the validation result is false, we compute a set of counterexamples for future training. This iteration is repeated until a trained candidate is validated or a given timeout is reached. The whole process of URS is sketched in Algorithm 1. LRS is achieved by replacing Eq. (8) and Eq. (10) with Eq. (9) and Eq. (11) in line 5, respectively.

Algorithm 1: Sketch of Upper Reward Martingales.

input : Perturbed System M_μ, Granularity τ, Refinement step length ξ.
output : Neural URS h or UNKNOWN.

1 $\tilde{S} \leftarrow \textit{Discretize}(S, \tau)$;
2 $h \leftarrow \textit{Initialize}(\cdot)$;
3 **while** *timeout not reached* **do**
4 $h \leftarrow \textit{Train}(h, \tilde{S})$; `// Train a candidate.`
5 **if** *Eq.* (8) ∧ *Eq.* (10) **then** `// Validate.`
6 Return h ; `// Return h once it is valid.`
7 **else**
8 $\tau \leftarrow \tau - \xi$; `// Make τ smaller.`
9 $\tilde{S} \leftarrow \textit{Discretize}(S, \tau) \cup \textit{Counterexamples}(h)$;
10 Return UNKNOWN.

Training Candidate Reward Martingales

The training phase involves two important steps, i.e., training data construction and loss function definition.

Discretizing Training Data Since the state space S is possibly continuous and infinite, to boost the training we choose a finite set of states and then train reward martingale candidates on it. This can be achieved by discretizing the state space S and constructing a *discretization* $\tilde{S} \subseteq S$ such that for each $s \in S$, there is a $\tilde{s} \in \tilde{S}$ with $\|s - \tilde{s}\|_1 < \tau$, where $\tau > 0$ is called the granularity of $\tilde{S}$. As S is compact and thus bounded, this discretization can be computed by simply picking vertices of a grid with sufficiently small cells. For the training after validation failure, $\tilde{S}$ is constructed on a set of counterexamples and a new finite set of states triggered by a smaller τ. Once the discretization $\tilde{S}$ is obtained, we construct three finite sets $S_{C1} := \tilde{S} \cap S_g$, $S_{C2} := \tilde{S} \cap \overline{S_g}$ and $S_{C3} := \tilde{S} \cap S_0$ used for the training process.

Loss Functions of URS A candidate URS is initialized as a neural network h_θ w.r.t. the network parameter θ. Then h_θ is learned by minimizing the following loss function:

$$\mathcal{L}_{URS}(\theta) := k_1 \cdot \mathcal{L}_{C1}(\theta) + k_2 \cdot \mathcal{L}_{C2}(\theta) + k_3 \cdot \mathcal{L}_{C3}(\theta) \qquad (3)$$

where k_i, $i = 1, 2, 3$ are the algorithm parameters balancing the loss terms.

The first loss term is defined via the boundedness condition of URS in Definition 3 as:

$$\mathcal{L}_{C1}(\theta) = \frac{1}{|S_{C1}|} \sum_{s \in S_{C1}} [\max\{h_\theta(s) - K', 0\} + \max\{K - h_\theta(s), 0\}] \qquad (4)$$

Intuitively, a loss will incur if either $h_\theta(s)$ is not bounded from above by K' or below by K for any $s \in S_{C1}$.

The second loss term is defined via the decreasing pre-expectation condition of URS in Definition 3 as:

$$\mathcal{L}_{C2}(\theta) = \frac{1}{|S_{C2}|} \sum_{s \in S_{C2}} \left(\max\{ \sum_{s' \in \mathcal{D}_s} \frac{h_\theta(s')}{N} - h_\theta(s) + \zeta, 0\} \right), \quad (5)$$

where for each $s \in S_{C2}$, $\mathcal{D}_s$ is the set of its successor states such that $\mathcal{D}_s := \{s' \mid s' = f(s, \pi(s+\delta_i)), \delta_i \sim \mu, i \in [1, N]\}$, $N > 0$ is the sample number of successor states. Note that h_θ is a neural network, so it is intractable to directly compute the closed form of its expectation. Instead, we use the mean of $h_\theta(\cdot)$ at the N successor states to approximate the expected value $\mathbb{E}_{\delta\sim\mu}[h(f(s, \pi(s+\delta)))]$ for each $s \in S_{C2}$, and ζ to tighten the decreasing pre-expectation condition. Details will be explained in Theorem 3.

The third loss term is the regularization term used to assure the tightness of upper bounds from URS:

$$\mathcal{L}_{C3}(\theta) := \frac{1}{|S_{C3}|} \sum_{s \in S_{C3}} (\max\{h_\theta(s) - \overline{u}, 0\}) \quad (6)$$

where $\overline{u}$ enforces the upper bounds always under some tolerable thresholds to make upper bounds as tight as possible.

Loss Functions of LRS Like URS, a candidate neural network LRS η_θ w.r.t. the parameter θ is learned by minimizing the loss function

$$\mathcal{L}_{LRS}(\theta) := k_1 \cdot \mathcal{L}_{C1'}(\theta) + k_2 \cdot \mathcal{L}_{C2'}(\theta) + k_3 \cdot \mathcal{L}_{C3'}(\theta) \quad (7)$$

where $k_i,\ i = 1, 2, 3$ are hyperparameters balancing the loss terms. $\mathcal{L}_{C1'}, \mathcal{L}_{C2'}$ are defined based on the LRS conditions in Definition 4, while $\mathcal{L}_{C3'}$ is the regularization term used to assure the tightness of lower bounds from LRS:

$$\mathcal{L}_{C1'}(\theta) = \frac{1}{|S_{C1}|} \sum_{s \in S_{C1}} \left(\max\{\eta_\theta(s) - K', 0\} + \max\{K - \eta_\theta(s), 0\} \right),$$

$$\mathcal{L}_{C2'}(\theta) = \frac{1}{|S_{C2}|} \sum_{s \in S_{C2}} \left(\max\{\eta_\theta(s) - \sum_{s' \in \mathcal{D}_s} \frac{\eta_\theta(s')}{N} - \zeta', 0\} \right),$$

$$\mathcal{L}_{C3'}(\theta) = \frac{1}{|S_{C3}|} \sum_{s \in S_{C3}} \left(\max\{\underline{l} - \eta_\theta(s), 0\} \right),$$

where ζ' is used to make the increasing pre-expectation condition stricter (see Theorem 3), and $\underline{l}$ is a hyper-parameter that enforces the lower bounds are as tight as possible by incentivizing not exceeding some tolerable thresholds.

Reward Martingale Validation

A candidate URS (resp. LRS) is validated if it meets the conditions in Definition 3 (resp. Definition 4). Because the candidate URS and LRS are neural networks, they are Lipschitz continuous (Ruan, Huang, and Kwiatkowska 2018). Thus, the condition in Definition 5 is satisfied straightforwardly. For the boundedness condition, we can check

$$\inf_{s \in S_g} h(s) \geq K \text{ and } \sup_{s \in S_g} h(s) \leq K' \quad (8)$$

or alternatively,

$$\inf_{s \in S_g} \eta(s) \geq K \text{ and } \sup_{s \in S_g} \eta(s) \leq K' \quad (9)$$

using the interval bound propagation approach (Gowal et al. 2018; Xu et al. 2020). When a state $s \in S_g$ violates the above conditions, it is treated as a counterexample and added to $\tilde{S}$ for future training.

For the decreasing and increasing pre-expectation conditions in Definitions 3 and 4, Theorem 3 establishes two corresponding sufficient conditions, which are easier to check.

Theorem 3. *Given an M_μ and a function $h : S \to \mathbb{R}$, we have $pre_h(s) \leq h(s)$ for any state $s \in \overline{S_g}$ if the formula below*

$$\mathbb{E}_{\delta\sim\mu}[h(f(\tilde{s}, \pi(\tilde{s}+\delta)))] \leq h(\tilde{s}) - \zeta \quad (10)$$

holds for any state $\tilde{s} \in \tilde{S} \cap \overline{S_g}$, where $\zeta = r_{\max} + \tau \cdot L_h \cdot (1 + L_f \cdot (1 + L_\pi))$ with L_f, L_π, L_h being the Lipschitz constants of f, π, h, and $r_{\max}$ being the maximum value of R, respectively. Analogously, we have $pre_\eta(s) \geq \eta(s)$ for any state $s \in \overline{S_g}$ if:

$$\mathbb{E}_{\delta\sim\mu}[\eta(f(\tilde{s}, \pi(\tilde{s}+\delta)))] \geq \eta(\tilde{s}) - \zeta' \quad (11)$$

holds for any state $\tilde{s} \in \tilde{S} \cap \overline{S_g}$, where $\zeta' = r_{\min} - \tau \cdot L_\eta \cdot (1 + L_f \cdot (1 + L_\pi))$ with $r_{\min}$ being the minimum value of R.

Similarly, any state violating Eqs. (10) and (11) is treated as a counterexample and will be added to $\tilde{S}$ for training.

To check the satisfiablility of Eqs. (10) and (11) in a state $\tilde{s}$, we need to compute the expected value $\mathbb{E}_{\delta\sim\mu}[h(f(\tilde{s}, \pi(\tilde{s}+\delta)))]$ and $\mathbb{E}_{\delta\sim\mu}[\eta(f(\tilde{s}, \pi(\tilde{s}+\delta)))]$. However, it is difficult to compute a closed form because h (resp.η) is provided in the form of neural networks. We devise two strategies below depending on the training approaches of control policies.

An Over-Approximation Approach For control policies that are trained on compact but infinitely continuous state space, we bound the expected value $\mathbb{E}_{\delta\sim\mu}[h(f(\tilde{s}, \pi(\tilde{s}+\delta)))]$ (resp. $\mathbb{E}_{\delta\sim\mu}[\eta(f(\tilde{s}, \pi(\tilde{s}+\delta)))]$) via interval arithmetic (Gowal et al. 2018; Xu et al. 2020) instead of computing it, which is inspired by the work (Lechner et al. 2022; Zikelic et al. 2023). In particular, given the noise distribution μ and its support $\mathrm{N} = \{\delta \in \mathbb{R}^n \mid \mu(\delta) > 0\}$, we first partition N into finitely k cells $\mathrm{cell(N)} = \{N_1, \cdots, N_k\}$, and use $\mathrm{maxvol} = max_{N_i \in \mathrm{cell(N)}} \mathrm{vol}(N_i)$ (resp. $\mathrm{minvol} = min_{N_i \in \mathrm{cell(N)}} \mathrm{vol}(N_i)$) to denote the maximal (resp. minimal) volume with respect to the Lebesgue measure of any cell in the partition, respectively. For the expected value in Eq. (10), we bound it from above:

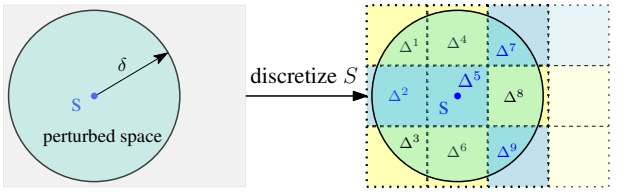
$$\mathbb{E}_{\delta\sim\mu}[h(f(\tilde{s}, \pi(\tilde{s}+\delta)))] \leq \sum_{N_i \in \mathrm{cell(N)}} \mathrm{maxvol} \cdot \sup_\delta F(\delta) \quad (12)$$

where $F(\delta) = h(f(\tilde{s}, \pi(\tilde{s}+\delta)))$. Similarly, for the expected value in Eq. (11), we bound it from below:

$$\mathbb{E}_{\delta\sim\mu}[\eta(f(\tilde{s}, \pi(\tilde{s}+\delta)))] \geq \sum_{N_i \in \mathrm{cell(N)}} \mathrm{minvol} \cdot \inf_\delta F(\delta) \quad (13)$$

Both supremum and infimum can be calculated via interval arithmetic. We refer interested readers to (Lechner et al. 2022) and (Zikelic et al. 2023) for more details.

By replacing the actual expected values with their overestimated upper bounds and lower bounds in Eqs. (12) and (13), the validation becomes pragmatically feasible without losing the soundness, i.e., a reward martingale candidate that is validated must be valid. However, due to the overestimation, it may produce false positives and incur unnecessary further training or even timeout.



Infinite and continuous perturbed state space

Discretized perturbed state space and the possibilities of actions

Figure 1: An example of state space discretization

An Analytic Approach We propose an analytical approach for the control policies that are trained on discretized abstract states. In previous work (Jin et al. 2022; Tian et al. 2022; Li et al. 2022), a compact but infinitely continuous state space S was discretized to a finite set of abstract states, i.e., $S = \bigcup_{i=1}^{L} S^i$ and $\forall i \neq j,\ S^i \cap S^j = \emptyset$. Then a neural network policy π was trained on the set of abstract states. After training, each abstract state S^i corresponds to a constant action a_i, i.e., $\pi(s) = a_i$ for all $s \in S^i, i = 1, \ldots, L$.

Because an abstract state space is finite, we can calculate the probabilities of all possible actions for the perturbed state $\hat{s} = s + \delta$ with $\delta \sim \mu$, i.e.,

$$\Delta^i := p(\hat{a} = a_i) = p(s + \delta \in S^i) \tag{14}$$

Fig. 1 shows an illustrative example. Because the system dynamics f is deterministic, the probability of the successor state s_i' equals to the probability of the action a_i, i.e., $p(f(s, \hat{a}) = s_i') = \Delta^i$. We can compute the analytic solution:

$$\mathbb{E}_{\delta\sim\mu}[h(f(s, \pi(s + \delta)))] = \sum_{i=1}^{L} h(s_i') \times \Delta^i, \tag{15}$$

where $\Delta^i = \int_{S^i} \mu'\, d\hat{s}$ with S^i being the set of all states whose action is a_i, and μ' being the distribution of the perturbed state $\hat{s}$ that obtained using the value of the actual state s and the noise distribution μ (Williams 1991). $\mathbb{E}_{\delta\sim\mu}[\eta(f(s, \pi(s + \delta)))]$ can be calculated in the same manner.

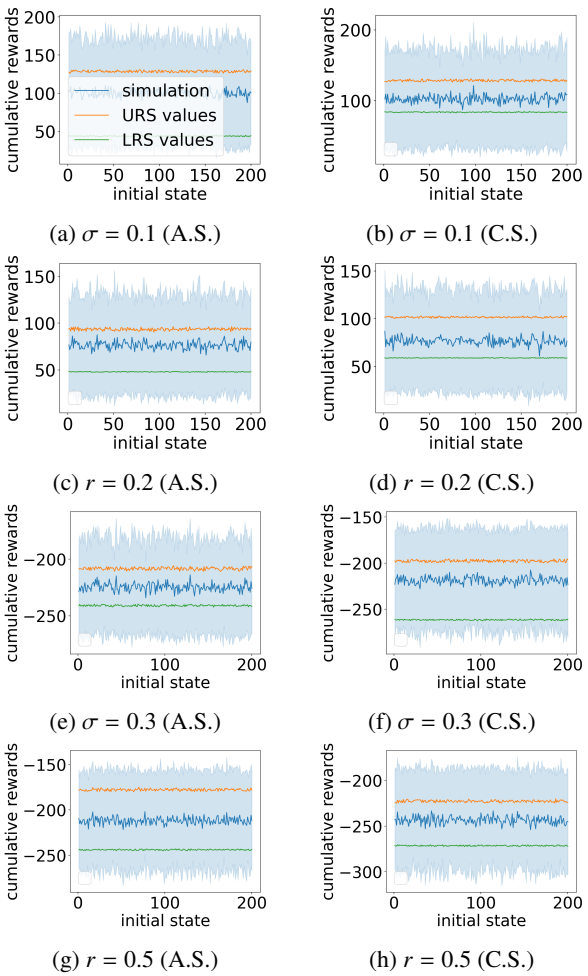
Experimental Evaluations

Our experimental objectives are to evaluate: (i) the effectiveness of certified upper and lower bounds, (ii) the effectiveness of certified tail bounds, and (iii) the efficiency of training and validating reward martingales.

Experimental Settings. We consider four benchmarking problems: CartPole (CP), MountainCar (MC), B1, and B2 from Gym (Brockman et al. 2016) and the benchmarks for reachability analysis (Ivanov et al. 2021), respectively. The finite termination and the concentration property of our examples are ensured by using existing methods in Lechner et al. (2022). To demonstrate the generality of our approach, we train systems with different activation functions and network structures of the planted NNs, using different DRL algorithms such as DQN (Mnih et al. 2013) and DDPG (Lillicrap et al. 2016). Table 1 gives the details of training settings.

Task	Dim.	Alg.	A.F.	Size	A.T.	S.P.	Training
CP	4	DQN	ReLU	3×200	Dis.	Gym	C.S./A.S.
MC	2	DQN	Sigmoid	2×200	Dis.	Gym	C.S./A.S.
B1	2	DDPG	ReLU	2×100	Cont.	R.A.	C.S./A.S.
B2	2	DDPG	Tanh	2×300	Cont.	R.A.	C.S./A.S.

Table 1: Experimental settings. (**Dim.**: dimension; **Alg.**: DRL algorithm; **A.F.**: activation function; **A.T.**: action type; **S.P.**: sources of problems; **Dis.**: discrete; **Cont.**: continuous; **R.A.**: reachability analysis; **C.S.**: training on concrete states; **A.S.**: training on abstract states.)



(a) $\sigma = 0.1$ (A.S.) (b) $\sigma = 0.1$ (C.S.)

(c) $r = 0.2$ (A.S.) (d) $r = 0.2$ (C.S.)

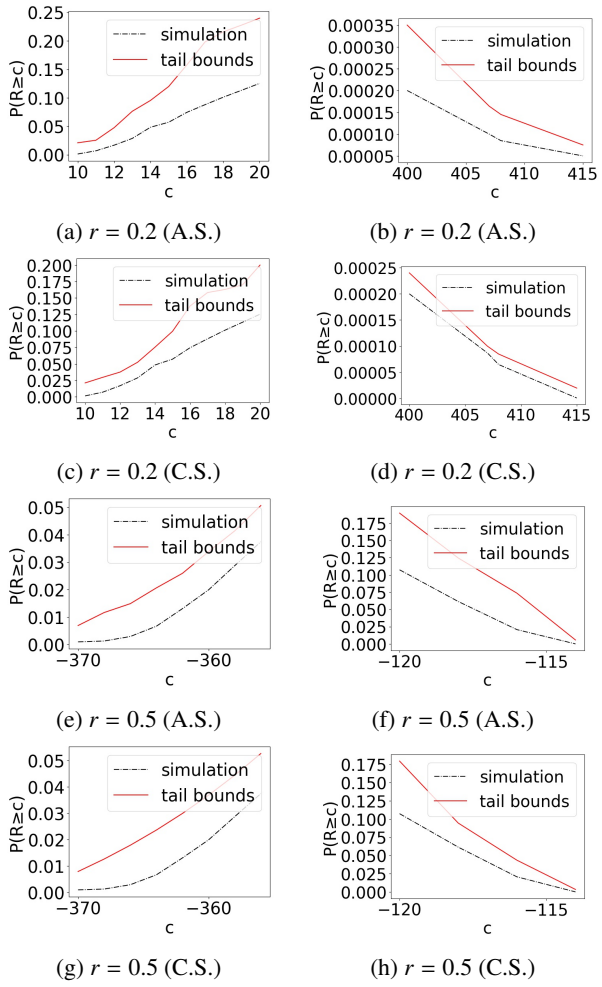
(e) $\sigma = 0.3$ (A.S.) (f) $\sigma = 0.3$ (C.S.)

(g) $r = 0.5$ (A.S.) (h) $r = 0.5$ (C.S.)

Figure 2: Certified bounds for CartPole (a-d) and B1 (e-h).

Effectiveness of Certified Upper and Lower Bounds. Fig. 2 shows the certified bounds of expected cumulative rewards (Theorem 1) and the simulation results for CP and B1 under different perturbations and policies, respectively. The x-axis indicates different initial states, while the y-axis means

For the robustness verification, we consider two different state perturbations, i.e., Gaussian noises with zero means and different deviations, and uniform noises with zero means and different radii. Specifically, for each state $s = (s_1, \ldots, s_n)$, we add noises $X_1, \ldots, X_n$ to each dimension of s and obtain the perturbed state $(s_1 + X_1, \ldots, s_n + X_n)$, where $X_i \sim \mathbf{U}(-r, r)$ $(1 \leq i \leq n)$ is some uniformly distributed noise or $X_i \sim \mathbf{G}(0, \sigma)$ $(1 \leq i \leq n)$ is some Gaussian distributed noise.



(a) $r = 0.2$ (A.S.) (b) $r = 0.2$ (A.S.)

(c) $r = 0.2$ (C.S.) (d) $r = 0.2$ (C.S.)

(e) $r = 0.5$ (A.S.) (f) $r = 0.5$ (A.S.)

(g) $r = 0.5$ (C.S.) (h) $r = 0.5$ (C.S.)

Figure 3: Certified tail bounds of expected cumulative rewards and simulation results for CartPole (a-d) and B1 (e-h).

the corresponding expected cumulative rewards. The orange lines represent the upper bounds calculated by the trained URSs, the green lines represent the lower bounds computed by the trained LRSs, and the blue lines and shadows represent the means and standard deviations of the simulation results that are obtained by executing 200 episodes for each initial state. We can find that the certified bounds tightly enclose the simulation outcomes, demonstrating the effectiveness of our trained reward martingales.

We also observe that the tightness of the certified bounds depends on particular systems, trained policies and perturbations, which is worth further investigating.

Effectiveness of Certified Tail Bounds. Fig. 3 depicts the certified tail bounds and statistical results for CP and B1 under different perturbations and policies. Due to the data sparsity of cumulative rewards, we choose 200 different initial states (instead of a single one) and execute the systems by 200 episodes for each initial state. We record the cumulative rewards, and the statistical results of tail probabilities for different c's are shown by the black dashed lines. For each initial state s_0 from the 200 initial states, we calculate $\mathbb{P}_{s_0}(\mathscr{R} \geq c)$ and $\mathbb{P}_{s_0}(\mathscr{R} \leq c)$ according to Theorem 2. Their average values $P(\mathscr{R} \geq c)$ and $P(\mathscr{R} \leq c)$ are shown by the red solid lines. The results show that our calculated tail bounds tightly enclose the statistical outcomes. We also observe that the trend of the calculated tail bounds is exponential upward or downward, which is consistent with Theorem 2.

			URS			LRS		
Task	**Pert.**	**Policy**	**T.T.**	**V.T.**	**To.T.**	**T.T.**	**V.T.**	**To.T.**
CP	$\sigma = 0.1$	A.S.	912	751	1663	1165	822	1987
		C.S.	1203	873	2076	1384	918	2302
	$r = 0.2$	A.S.	901	409	1310	1081	535	1616
		C.S.	898	529	1427	1009	649	1658
B1	$\sigma = 0.3$	A.S.	501	45	546	629	41	670
		C.S.	570	53	623	850	189	1036
	$r = 0.5$	A.S.	530	47	577	681	175	856
		C.S.	593	54	647	945	280	1225

Table 2: Time on training and validating reward martingales. (**T.T.**: training time; **V.T.**: validating time; **To.T.**: total time.)

Efficiency Comparison. Table 2 shows the time cost of training and validating processes under different policies and perturbations. In general, training costs more time than validating, and high-dimensional systems e.g., CP (4-dimensional state space) cost more time than low-dimensional ones, e.g, B1 (2-dimensional state space). That is because the validation step suffers from the curse of high-dimensionality (Berkenkamp et al. 2017).

We also observe that, the analytic approach is more efficient in training and validating than the over-approximation-based approach. This result is consistent to the fact that the analytical approach can produce more precise results of the expected values and consequently can reduce false positives and unnecessary refinement and training. The conclusions from the above results are also applicable to MC and B2.

Concluding Remarks

In this paper, we have introduced a groundbreaking quantitative robustness verification approach for perturbed DRL-based control systems, utilizing the innovative concept of reward martingales. Our work has established two fundamental theorems that serve as cornerstones: the certification of reward martingales as rigorous upper and lower bounds, as well as their role in tail bounds for system robustness. We have presented an algorithm that effectively trains reward martingales through the implementation of neural networks. Within this algorithm, we have devised two distinct methods for computing expected values, catering to control policies developed across diverse state space configurations. Through an extensive evaluation encompassing four classical control problems, we have convincingly showcased the versatility and efficacy of our proposed approach.

We believe this work would inspire several future studies to reduce the complexity in validating reward martingales for high dimensional systems. Besides, it is also worth further investigation to advance the approach for calculating tighter certified bounds by training more precise reward martingales.

Acknowledgments

The work has been supported by the NSFC-ISF Joint Program (62161146001, 3420/21), NSFC Project (62372176, 62306116), Huawei Technologies Co., Ltd., Shanghai International Joint Lab of Trustworthy Intelligent Software (Grant No. 22510750100), Shanghai Trusted Industry Internet Software Collaborative Innovation Center, the Engineering and Physical Sciences Research Council (EP/T006579/1), National Research Foundation (NRF-RSS2022-009), Singapore, and Shanghai Jiao Tong University Postdoc Scholarship.

References

Abate, A.; Giacobbe, M.; and Roy, D. 2021. Learning Probabilistic Termination Proofs. In *CAV*, volume 12760, 3–26.

Bacci, E. 2022. *Formal verification of deep reinforcement learning agents*. Ph.D. thesis, University of Birmingham, UK.

Berkenkamp, F.; Turchetta, M.; Schoellig, A. P.; and Krause, A. 2017. Safe Model-based Reinforcement Learning with Stability Guarantees. In *NeurIPS*, 908–918.

Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. OpenAI Gym. ArXiv:1606.01540.

Chakarov, A.; and Sankaranarayanan, S. 2013. Probabilistic Program Analysis with Martingales. In *CAV*, 511–526.

Chen, C.; Wei, H.; Xu, N.; Zheng, G.; Yang, M.; Xiong, Y.; Xu, K.; and Li, Z. 2020. Toward A Thousand Lights: Decentralized Deep Reinforcement Learning for Large-Scale Traffic Signal Control. In *AAAI*, 3414–3421.

Corsi, D.; Marchesini, E.; and Farinelli, A. 2021. Formal verification of neural networks for safety-critical tasks in deep reinforcement learning. In *UAI*, volume 161, 333–343.

Dawson, C.; Gao, S.; and Fan, C. 2023. Safe Control With Learned Certificates: A Survey of Neural Lyapunov, Barrier, and Contraction Methods for Robotics and Control. *IEEE Trans. Robotics*, 39(3): 1749–1767.

Deshmukh, J. V.; and Sankaranarayanan, S. 2019. Formal techniques for verification and testing of cyber-physical systems. In *Design Automation of Cyber-Physical Systems*, 69–105.

Drews, S.; Albarghouthi, A.; and D'Antoni, L. 2020. Proving data-poisoning robustness in decision trees. In *PLDI*, 1083–1097.

Gowal, S.; Dvijotham, K.; Stanforth, R.; Bunel, R.; Qin, C.; Uesato, J.; Arandjelovic, R.; Mann, T. A.; and Kohli, P. 2018. On the Effectiveness of Interval Bound Propagation for Training Verifiably Robust Models. *CoRR*, abs/1810.12715.

Hoeffding, W. 1994. Probability inequalities for sums of bounded random variables. *The collected works of Wassily Hoeffding*, 409–426.

Ivanov, R.; Carpenter, T.; Weimer, J.; Alur, R.; Pappas, G.; and Lee, I. 2021. Verisig 2.0: Verification of neural network controllers using taylor model preconditioning. In *CAV*, 249–262.

Jin, P.; Tian, J.; Zhi, D.; Wen, X.; and Zhang, M. 2022. Trainify: A CEGAR-Driven Training and Verification Framework for Safe Deep Reinforcement Learning. In *CAV*, volume 13371, 193–218.

Kumar, A.; Levine, A.; and Feizi, S. 2022. Policy Smoothing for Provably Robust Reinforcement Learning. In *ICLR*.

Larsen, K.; Legay, A.; Nolte, G.; Schlüter, M.; Stoelinga, M.; and Steffen, B. 2022. Formal methods meet machine learning (F3ML). In *ISoLA*, 393–405.

Lechner, M.; Zikelic, D.; Chatterjee, K.; and Henzinger, T. A. 2022. Stability Verification in Stochastic Control Systems via Neural Network Supermartingales. In *AAAI*, 7326–7336.

Li, Z.; Zhu, D.; Hu, Y.; Xie, X.; Ma, L.; Zheng, Y.; Song, Y.; Chen, Y.; and Zhao, J. 2022. Neural Episodic Control with State Abstraction. In *ICLR*.

Lillicrap, T. P.; Hunt, J. J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; and Wierstra, D. 2016. Continuous control with deep reinforcement learning. In *ICLR*.

Liu, B.; and Ding, Z. 2022. A distributed deep reinforcement learning method for traffic light control. *Neurocomputing*, 490: 390–399.

Lütjens, B.; Everett, M.; and How, J. P. 2019. Certified Adversarial Robustness for Deep Reinforcement Learning. In *CoRL*, volume 100, 1328–1337.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. A. 2013. Playing Atari with Deep Reinforcement Learning. *CoRR*, abs/1312.5602.

Oikarinen, T. P.; Zhang, W.; Megretski, A.; Daniel, L.; and Weng, T. 2021. Robust Deep Reinforcement Learning through Adversarial Loss. In *NeurIPS*, 26156–26167.

Ruan, W.; Huang, X.; and Kwiatkowska, M. 2018. Reachability Analysis of Deep Neural Networks with Provable Guarantees. In *IJCAI*, 2651–2659.

Tian, J.; Zhi, D.; Liu, S.; Wang, P.; Katz, G.; and Zhang, M. 2022. BBReach: Tight and Scalable Black-Box Reachability Analysis of Deep Reinforcement Learning Systems. *CoRR*, abs/2211.11127.

Wan, X.; Sun, M.; Chen, B.; Chu, Z.; and Teng, F. 2023. AdapSafe: Adaptive and Safe-Certified Deep Reinforcement Learning-Based Frequency Control for Carbon-Neutral Power Systems. In *AAAI*, 5294–5302.

Wan, X.; Zeng, L.; and Sun, M. 2022. Exploring the Vulnerability of Deep Reinforcement Learning-based Emergency Control for Low Carbon Power Systems. In *IJCAI*, 3954–3961.

Wang, P. 2022. Tail-Bound Cost Analysis over Nondeterministic Probabilistic Programs. *Journal of Shanghai Jiaotong University (Science)*, 1–11.

Wang, P.; Fu, H.; Goharshady, A. K.; Chatterjee, K.; Qin, X.; and Shi, W. 2019. Cost analysis of nondeterministic probabilistic programs. In *PLDI*, 204–220.

Williams, D. 1991. *Probability with martingales*. Cambridge university press.

Xu, K.; Shi, Z.; Zhang, H.; Wang, Y.; Chang, K.; Huang, M.; Kailkhura, B.; Lin, X.; and Hsieh, C. 2020. Automatic Perturbation Analysis for Scalable Certified Robustness and Beyond. In *NeurIPS*.

Zhang, H.; Chen, H.; Xiao, C.; Li, B.; Liu, M.; Boning, D. S.; and Hsieh, C. 2020. Robust Deep Reinforcement Learning against Adversarial Perturbations on State Observations. In *NeurIPS*, 21024–21037.

Zhang, H.; Gu, J.; Zhang, Z.; Du, L.; Zhang, Y.; Ren, Y.; Zhang, J.; and Li, H. 2023. Backdoor attacks against deep reinforcement learning based traffic signal control systems. *Peer Peer Netw. Appl.*, 16(1): 466–474.

Zhang, W.; Tu, Z.; and Liu, W. 2023. Optimal Charging Control of Energy Storage Systems for Pulse Power Load Using Deep Reinforcement Learning in Shipboard Integrated Power Systems. *IEEE Trans. Ind. Informatics*, 19(5): 6349–6363.

Zikelic, D.; Lechner, M.; Henzinger, T. A.; and Chatterjee, K. 2023. Learning Control Policies for Stochastic Systems with Reach-Avoid Guarantees. In *AAAI*, 11926–11935.