Learning to Act Safely with Limited Exposure and Almost Sure Certainty

Agustin Castellano, Hancheng Min, Juan Bazerque, and Enrique Mallada

Abstract— **This paper aims to put forward the concept that learning to take safe actions in unknown environments, even with probability one guarantees, can be achieved without the need for an unbounded number of exploratory trials, provided that one is willing to navigate trade-offs between optimality, level of exposure to unsafe events, and the maximum detection time of unsafe actions. We illustrate this concept in two complementary settings. We first focus on the canonical multi-armed bandit problem and seek to study the intrinsic trade-offs of learning safety in the presence of uncertainty. Under mild assumptions on sufficient exploration, we provide an algorithm that provably detects all unsafe machines in an (expected) finite number of rounds. The analysis also unveils a trade-off between the number of rounds needed to secure the environment and the probability of discarding safe machines. We then consider the problem of finding optimal policies for a Markov Decision Process (MDP) with almost sure constraints. We show that the (action) value function satisfies a barrier-based decomposition which allows for the identification of feasible policies independently of the reward process. Using this decomposition, we develop a Barrier-learning algorithm, that identifies such unsafe state-action pairs in a finite expected number of steps. Our analysis further highlights a trade-off between the time lag for the underlying MDP necessary to detect unsafe actions, and the level of exposure to unsafe events. Simulations corroborate our theoretical findings, further illustrating the aforementioned trade-offs, and suggesting that safety constraints can further speed up the learning process.**

Index Terms— **Uncertain systems, randomized algorithms, Markov processes, iterative learning control, optimal control**

I. Introduction

Motivated by the success of machine learning in achieving super human performance, e.g., in vision [2], speech [3], and video games [4], there has been recent interest in developing learning-enabled technology that can implement highly complex actions for safety-critical autonomous systems, such as self-driving cars, robots, etc. However, without proper safety guarantees such systems will rarely be deployed. There is therefore the need to develop analysis tools and algorithms that can provide such guarantees during, and after, training. Efforts to provide such guarantees can be broadly grouped in two lines of work with somehow complementary success.

The first approach leverages model-based techniques, based on Lyapunov stability [5] and robust control [6], to provide (worst-case) safety guarantees based on a nominal model and assumptions on uncertainty and disturbances [7]–[17]. In such settings, safety is usually specified in terms of stability, robust stability, or the existence of some invariant or control invariant sets. Notably, such methods provide highly reliable guarantees, allowing the system to operate without ever violating safety specifications. However, they require an a priori fixed specification of the safety notion, e.g., a set or stability, that is implicitly encoded by the controller/agent. Moreover, due to the worst case approach to uncertainty, these methods tend to suffer poor performance in the average case.

The second line of work, in which our work naturally lies in, seeks to provide safety guarantees in model-free settings by adding constraints to the learning problem [18]–[29]. In this way, safety specifications can be further extended, beyond typical control notions, at the expense of introducing uncertainty, and risk, in the safety guarantees. This lack of certainty is usually mitigated in different ways. For example, one can introduce soft constraints to primal methods [19], [24], or use primal-dual methods [25], [26], that lead to asymptotic constraint satisfaction. Such methods can indeed provide asymptotic a.s. guarantees, but at the expense of incurring unbounded violations of the constraints during training [26], an undesired feature for most control problems. A different approach consists on introducing an initial stage in the algorithm, to be executed before the optimization starts, that aims at learning an approximation of the safety region [22], [28]. This approach naturally leads to high probability guarantees on constraint satisfaction (based on Probably Approximately Correct (PAC) sample complexity bounds). However, due to the need to limit the running time of the safety assessment stage, it is impossible to provide probability one guarantees, which can in turn lead to catastrophic consequences.

The above discussion implicitly illustrates a complementary set of challenges that one needs to overcome to develop reliable model-free learning algorithms for safety critical applications.

- *Exposure to Failures.* Due to the desire for limited constraint violations, one is required to perform the learning task with limited information of the unsafe region.
- *Outcome Uncertainty.* Along the same line, making decisions with finite number of samples intrinsically requires one to accept the possibility of making errors.
- *Assessment with Deferred Consequences.* The real-time nature of these problems leads to settings where information is revealed sequentially and in a delayed manner.

The goal of this paper is to develop a novel approach

A preliminary version of this work was previously presented in [1].

A. Castellano and J. Bazerque are with the Department of Electrical Engineering at Universidad de la República, Uruguay (e-mail: {acastellano, jbazerque}@fing.edu.uy).

H. Min and E. Mallada are with the Department of Electrical and Computer Engineering at Johns Hopkins University, Baltimore, MD 21218 USA (e-mail: {hanchmin,mallada}@jhu.edu).

for model-free safety assessment, considering the complementary benefits of the aforementioned techniques, with the aim of overcoming these challenges. To achieve this goal, we argue and demonstrate that, due to the logical (safe/unsafe) nature of safety assessment, *the problem of finding safe actions is fundamentally different from that of finding the best action.* As a result, safety assessment can be achieved more efficiently and independently of the optimization task. This motivates the need of a separate (logical) signalling mechanism–akin to the reward in multi-armed bandits (MAB) [30] and reinforcement learning (RL) [31]–dedicated to inform, either directly or indirectly, about a violation of the safety specification. We call such signal here a *damage indicator*, denoted by $D_t \in \{0,1\}$.

Using this newly introduced safety signal D_t we aim at developing safety assessment algorithms that lead to actions A_t that satisfy a safety specifications of the from $D_t = 0$ a.s. or $P(D_t = 1) \leq \mu$ (conditioned on the action) for some nominal safety margin $\mu > 0$. Our approach leads to algorithms that can be implemented *sequentially* and *concurrently* with other optimization algorithms (such as Q-learning [32] or its variants [33], [34]), can achieve *probability one* guarantees on the safety specification, and only incur a *finite number of constraint violations*. Achieving such remarkable outcome naturally requires to navigate trade-offs across multiple requirements. We illustrate such trade-off by developing safety assessment methods for two exemplary cases.

We first consider the problem of safety assessment in MABs, where the safety specification is given by some $\mu \geq 0$ (Section II). While the case $\mu = 0$ is rather straightforward (Section II-A), when $\mu > 0$ requires to trade-off between quickly discarding unsafe arms and accurately estimating $P(D_t = 1|A_t)$. We thus make the conscious choice of focusing on prioritizing the rapid (finite time) detection of unsafe actions almost surely, which naturally requires to (mildly) give up optimality by possibly discarding some safe arms (Section II-B). We then move towards the problem of RL for Constrained Markov Decision Processes (Section III), wherein we seek to develop algorithms that can rapidly discard unsafe state-action pairs. In this setting decisions can have safety consequences that may only be realized after some time. By focusing on the almost sure safety specification ($\mu = 0$), we develop a decomposition framework (Section III-A), based on hard barrier functions, that allows to decouple the safety assessment problem from the problem of maximizing rewards. This leads to a novel barrier learner algorithm, that is able to identify all state-action pairs that lead to unsafe events (Section III-B). Our analysis further shows the explicit role that the delayed consequences have in the learning process (Section III-C). Numerical illustrations, in Section IV verify our theoretical analysis and further suggest that can further aid in the learning process.

II. Multi-armed bandits

We consider the setting of a stochastic bandit problem, with K arms indexed as $a \in \{1,\dots,K\}$. In the standard bandit problem an agent aims to devise an arm-pulling policy to optimize a reward. Here, we switch focus to the safety problem, for which we consider that pulling some the arms could be unsafe and lead to system damage or harm to the agent. Specifically, at each round $t \geq 1$ the agent pulls an arm A_t and obtains a binary-valued *damage indicator* D_t. If D_t is zero (one) this means that the action led to a safe (unsafe) result. We have that $\mathbb{E}[D_t|A_t] = \mu_{A_t}$. Each machine is therefore characterized by its *safety parameter* μ_a. The greater this value is, the more likely it is that pulling the machine will lead to an unsafe event. The goal of the player in this setup is to identify all the machines that are μ-unsafe, which is hereby defined.

Definition 1. *Given a* safety specification $\mu \in [0,1)$*, a machine a is said to be μ-unsafe if and only if $\mu_a > \mu$. Accordingly, a machine is μ-safe whenever $\mu_a \leq \mu$.*

The safety requirement μ is a design parameter, and is the only data that the player has access to along with the signal D_t. We will look at two distinct cases:

A) *Flawless setting* ($\mu = 0$): in this setting we only allow for flawless machines, and therefore would like to identify every machine whose probability of damage is non-zero (albeit very small).
B) *Relaxed setting* ($\mu > 0$): In this setting we want to identify unsafe machines with $\mu_a > \mu$. This means that we allow "somewhat defective" machines.

We will focus first on the flawless setting, as it will allow us to build intuition on how to devise a proper Algorithm and on the values of metrics involved. The solution in this case is straightforward: let the agent pull each arm and avoid arms that have led to an unsafe event $D_t = 1$. For the second case, we will rely on building a one-sided Sequential Probability Ratio Test (SPRT) [35], that will make us try each machine a sufficient number of times; if the machine is *unsafe*, the test will eventually decide on that hypothesis.

In both cases, our goal is the same. We want to *detect* all the machines that are unsafe. To that end, let us define at each round $t \geq 1$ the candidate safe set $\mathcal{A}_t$, which contains all the arms that haven't been classified as unsafe.

Assumption 1. *Given a safety requirement μ, there are M μ-unsafe machines, where $1 \leq M \leq K$. Without loss of generality we will assume the the first M arms to be μ-unsafe (i.e. $\mu_a > \mu,\ a = 1,\dots,M$)*

Definition 2. *For each round $t \geq 1$ we define the* candidate safe set $\mathcal{A}_t$ *as the set containing all the arms that have not been classified as unsafe.*

Set $\mathcal{A}_t$ is initialized as $\mathcal{A}_0 = \{1,\dots K\}$, and sequentially trimmed down whenever an arm is found to be unsafe. To select which arm to pull at each round, we consider probability distribution ψ, and select actions according to $A_t \sim \psi(\mathcal{A}_t)$.

Although we recognize that detecting unsafe machines necessarily implies pulling from those unsafe arms, we want to have a notion of whether our decision rules choose machines in an efficient manner. It is with that goal in mind that we define at each round the *exposure*.

Definition 3. *For $t \geq 1$ we define the* exposure *at round t as*

$$E_t = \sum_{\tau=1}^{t} \mathbb{1}\left(\mu_{A_\tau} > \mu\right). \tag{1}$$

This metric counts the rounds in which μ-unsafe machines have been pulled, regardless of whether they led to an unsafe event or not. Notice that the exposure inherits the randomness of the sequence of decisions A_t. Our results throughout this section will deal then with the expected value $\mathbb{E}[E_t]$. Ideally a good player would be one that attains low exposure—meaning it selects unsafe machines infrequently.

Remark 1. *We define the notion of* exposure *in the sense of "the condition of being unprotected" as its definition states. This marks a stark contrast with the notion of* regret, *typically studied in Bandit settings [30], where unbounded regret is unavoidable [36].*

At time t, the number of times an arm a has been pulled is

$$N_a(t) = \sum_{\tau=1}^{t} \mathbb{1}(A_\tau = a). \tag{2}$$

The following lemma states that the expected exposure coincides with the sum of the expected number of pulls over the unsafe machines.

Lemma 1. $\mathbb{E}[E_t] = \sum_{a=1}^{M} \mathbb{E}[N_a(t)]$

Proof.

$$\mathbb{E}[E_t] = \sum_{\tau=1}^{t} \mathbb{E}\left[\mathbb{1}\left(\mu_{A_\tau} > \mu\right)\right] = \sum_{\tau=1}^{t}\sum_{a=1}^{K} P(A_\tau = a)\mathbb{1}\left(\mu_a > \mu\right)$$
$$= \sum_{\tau=1}^{t}\sum_{a=1}^{M} P(A_\tau = a) = \sum_{a=1}^{M}\sum_{\tau=1}^{t} \mathbb{E}\left[\mathbb{1}(A_\tau = a)\right] = \sum_{a=1}^{M} \mathbb{E}[N_a(t)]$$

□

Definition 4. *The* conservation ratio $C_{\varepsilon,t}$ *is the proportion of safe machines kept at time t*

$$C_{\varepsilon,t} = \frac{|\mathcal{A}_t \cap \mathcal{A}^\star_\varepsilon|}{|\mathcal{A}^\star_\varepsilon|} \tag{3}$$

where $\mathcal{A}_t$ is the candidate safe set and $\mathcal{A}^\star_\varepsilon$ is the set containing all arms that are $(\mu - \varepsilon)$-safe: $\mathcal{A}^\star_\varepsilon = \{a \in \mathcal{A} : \mu_a \leq \mu - \varepsilon\}$, where $0 \leq \varepsilon \ll 1$ is a non-negative slack parameter.

This ratio gives the proportion of $(\mu - \varepsilon)$-safe machines present in the candidate safe set at each time step. ($C_{\varepsilon,t}$ close to 1 is desirable). The need for the conservativeness given by ε is that we want to detect unsafe machines in finite time. This will become clearer when we discuss the *relaxed setting*, for now it suffices to assume $\varepsilon = 0$.

A. Flawless setting ($\mu = 0$)

We start with the simplest case imaginable, which is that of a rigorous safety requirement of $\mu = 0$. In this setting, any machine that has positive probability of giving damage $D_t = 1$ should be deemed unsafe. The strategy for discarding these machines is pretty straightforward: at each round t select an arm A_t following strategy $\psi(\mathcal{A}_t)$ and, if the resulting damage is $D_t = 1$, classify the machine as unsafe by taking it out of the candidate safe set $\mathcal{A}_t$. This decision rule, summarized in Algorithm 1 has two interesting properties: *i)* all unsafe machines are eventually found, and *ii)* no safe machines are discarded along the way, as the the following theorem states.

Algorithm 1: Flawless Inspector

```
Input: Number of arms $K$, strategy $\psi$
/* Initialize candidate safe set */
$\mathcal{A}_0 = \{1, \dots, K\}$
for $t = 1, 2, \dots,$ do
    Pick arm $A_t \sim \psi(\mathcal{A}_t)$
    Observe damage $D_t$
    if $D_t = 1$ then
        /* trim unsafe arm */
        $\mathcal{A}_t \leftarrow \mathcal{A}_{t-1} \setminus \{A_t\}$
    end
end
```

Theorem 1. *Under Algorithm 1, for every strategy ψ, the following (in)equalities hold with probability 1 for all $t \geq 1$:*

$$\mathbb{E}[C_{0,t}] = 1 \tag{4}$$

$$\mathbb{E}[E_t] \leq \sum_{a=1}^{M} \frac{1}{\mu_a} \tag{5}$$

Proof. The proof for the safety ratio $C_{0,t}$ is immediate, since Algorithm 1 can never discard a safe machine (the event $D_t = 0$ has zero probability when pulling from a flawless arm). For the remaining equalities, let N_a be the number of pulls of the a-th arm needed to classify it as unsafe, which is well-defined for $a = 1, \dots, M$. We have that $N_a \sim \text{Geometric}(\mu_a)$, hence

$$\mathbb{E}[N_a] = \sum_{n=1}^{\infty} P(N_a = n)n = \sum_{n=1}^{\infty} \mu_a(1-\mu_a)^{n-1}n = \frac{1}{\mu_a} \tag{6}$$

Furthermore, for all t we have $N_a(t) \leq N_a$. Taking expectation on both sides and using (6) yields $\mathbb{E}[N_a(t)] \leq \frac{1}{\mu_a}$. Combining this with the result of Lemma 1 gives (5). □

We now state that under a uniform strategy, Algorithm 1 finds all unsafe machines in expected finite time.

Theorem 2. *Consider the uniform strategy $\psi_{\texttt{unif}}$ which at each round t samples uniformly from the candidate safe set $\mathcal{A}_t$. Let T be the time it takes for Algorithm 1 to detect all the unsafe machines under $\psi_{\texttt{unif}}$. Then*

$$\mathbb{E}[T] \leq \frac{K}{\mu_{\text{low}}}\left(\sum_{k=1}^{M}\frac{1}{k}\right),$$

where $\mu_{\text{low}} \leq \mu_a,\ \forall a = 1, \dots, M$.

Proof. The proof is in the Appendix A. □

Remark 2. *The uniform strategy ($\psi_{\texttt{unif}}$) in Theorem 2 is only assumed to simplify exposition. A finite bound on $\mathbb{E}[T]$ can still be obtained whenever $\psi_{\texttt{unif}}$ is replaced by any strategy that samples all actions from the set $\mathcal{A}_t$ with positive probability.*

B. Relaxed setting ($\mu > 0$)

We next consider the *relaxed* setting, in which we allow machines that give damage D_t with (possibly low) probability μ. This means that in order to identify unsafe machines we can no longer discard them at first sign of damage, but must rather pull from each arm and observe multiple unsafe events in order to be confident that the machine in question is defective. To check whether an arm a is defective or not we build the following hypothesis test $\mathbb{H}_a$

$$(\mathbb{H}_a)\ \begin{cases}\mathcal{H}_0: & \mu_a \le \mu - \varepsilon \\ \mathcal{H}_1: & \mu_a > \mu\end{cases} \tag{7}$$

in which the alternative hypothesis is that the machine is μ-unsafe, and where we introduce the slack parameter $\varepsilon \in (0, \mu]$. In order to solve (7), we will devise a Sequential Probability Ratio Test (SPRT) which is based on Abraham Wald's seminal work [35] with the following properties:

1) If the machine is unsafe —meaning $\mathcal{H}_1$ is true— the test will terminate in expected finite pulls $\mathbb{E}[N_a]$.
2) If the machine is safe, the probability that the test —incorrectly— decides on $\mathcal{H}_1$ is upper bounded by α, where $\alpha \in (0, 1)$ is the *failure tolerance* of the test.
3) Lowering failure tolerance α necessarily implies more pulls N_a to detect unsafe machines.
4) If $\mu_a \in (\mu - \varepsilon, \mu)$ the test is inconclusive.
5) The test is one-sided: it only decides on $\mathcal{H}_1$. Similar to Algorithm 1, whenever a machine is classified as unsafe it is not pulled any longer.

For a fixed arm a, let $\mathbf{d}_a(t) = \{D_\tau : A_\tau = a, \tau \le t\}$ be the (binary) sequence of outcomes of the a-th machine up to time t. The sequential probability ratio test relies on computing the log-likelihood ratio at each time step:

$$\Lambda_a(t) = \log \frac{f_{\mu-\varepsilon}(\mathbf{d}_a(t))}{f_\mu(\mathbf{d}_a(t))}\ , \tag{8}$$

where f_μ and $f_{\mu-\varepsilon}$ are the likelihood that the sequence $\mathbf{d}_a(t)$ under $\mathcal{H}_0$), and $\mathcal{H}_1$, respectively. The test terminates by declaring $\mathcal{H}_1$ whenever

$$\Lambda_a(t) \ge \log(1/\alpha)\ . \tag{9}$$

By means of sufficient statistics, $\Lambda_a(T)$ can be written as a function of both k, the total number of outcomes of $D_t = 1$ and $N_a(T)$, the total number of pulls up to time T. Considering a single-machine, the testing procedure is as follows. For each round $t \ge 1$ pull the arm and (given μ and ε) update the log-likelihood in (8). If (9) holds, then terminate the test, otherwise observe another sample D_t and repeat.

The following two lemmas state the desired behavior of the SPRT. Namely, that *i)* if the machine is unsafe, the SPRT will declare $\mathcal{H}_1$ with probability 1, *ii)* if the machine is safe, the SPRT will (incorrectly) declare $\mathcal{H}_1$ with probability less than or equal to α, and *iii)* the time of detection for unsafe machines is finite in expectation, and is well characterized in terms of the design parameters μ, ε and α.

Lemma 2. *For a fixed arm a of parameter μ_a, consider the sequential probability ratio test defined by* (7)–(9)*, where μ, ε and α are given. Then:*

i) If $\mathcal{H}_1$ is true, the test will (correctly) declare $\mathcal{H}_1$ with probability 1.
ii) If $\mathcal{H}_0$ is true, the test will keep going indefinitely with probability $\ge 1 - \alpha$

Proof. The proof is in the Appendix B. □

Lemma 3. *For a fixed arm a of parameter μ_a, consider the sequential probability ratio test defined by* (7)–(9)*, where μ, ε and α are given. Then, if the alternative $\mathcal{H}_1$ is true, the test is expected to terminate after T_a steps, with*

$$\mathbb{E}[T_a] \le 1 + \frac{\log(1/\alpha)}{\mathrm{kl}(\mu, \mu - \varepsilon)}\ , \tag{10}$$

where $\mathrm{kl}(\mu, \mu-\varepsilon)$ *is the Kullback-Leibler divergence between Bernoulli distributions*

$$\mathrm{kl}(\mu, \mu - \varepsilon) = \mu \log \frac{\mu}{\mu - \varepsilon} + (1 - \mu) \log \frac{1 - \mu}{1 - \mu + \varepsilon}\ . \tag{11}$$

Proof. The proof is in the Appendix B. □

Remark 3. *The preceeding lemma elucidates the need of the slack parameter ε: separating the two limiting distributions enables termination of the test under $\mathcal{H}_1$ in finite time. It also unveils two fundamental trade-offs. Firstly, if the distance between the limiting distributions increases (by enlarging ε), then the test detects unsafe machines faster. However, it becomes inconclusive over a larger proportion of the machines, since nothing can be assured in the region $\mu_a \in (\mu - \varepsilon, \mu)$. Secondly, α can be increased in order to detect unsafe machines faster, though this comes at the cost of declaring $\mathcal{H}_1$ for a larger proportion of the safe machines.*

We build Algorithm 2 based on this SPRT, and state its main properties in the following theorem.

Algorithm 2: Relaxed Inspector

```
Input: Number of arms $K$, strategy $\psi$
requirement $\mu$, slack $\varepsilon$, tolerance $\alpha$
/* Init. safe set and ratios */
$\mathcal{A}_0 = \{1, \dots, K\}$, $\Lambda_a = 0\ \ \forall a = 1, \dots, K$
for $t = 1, 2 \dots$ do
    Pick arm $A_t \sim \psi(\mathcal{A}_t)$
    Observe damage $D_t$
    /* Update log-likelihood ratio */
    $\Lambda_{A_t} \leftarrow \Lambda_{A_t} + \log \frac{f_\mu(D_t)}{f_{\mu-\varepsilon}(D_t)}$
    if $\Lambda_{A_t} \ge \log(1/\alpha)$ then
        /* SPRT terminates, trim unsafe
           arm */
        $\mathcal{A}_t \leftarrow \mathcal{A}_{t-1} \setminus \{A_t\}$
    end
end
```

Theorem 3. *Under Algorithm 2, for every strategy ψ, the following inequalities hold with probability 1 for all t:*

$$\mathbb{E}[C_{\varepsilon,t}] \geq 1-\alpha \tag{12}$$

$$\mathbb{E}[E_t] \leq M\left(1+\frac{\log(1/\alpha)}{\mathrm{kl}(\mu,\mu-\varepsilon)}\right) \tag{13}$$

where $\mathrm{kl}(\mu,\mu-\varepsilon)$ is the Kullback-Leibler divergence between Bernoulli distributions (11).

Proof. The proof follows from Lemma 2 and Lemma 3. □

In the same spirit as for the flawless setting, we now state that under a uniform strategy Algorithm 2 finds all unsafe machines in expected finite time.

Theorem 4. *Consider the uniform strategy $\psi_{\texttt{unif}}$ which at each round t samples uniformly from the candidate safe set $\mathcal{A}_t$. Let T be the time it takes for Algorithm 2 to detect all the unsafe machines under $\psi_{\texttt{unif}}$. Then*

$$\mathbb{E}[T] \leq M(K-M+1)\left(1+\frac{\log(1/\alpha)}{\mathrm{kl}(\mu,\mu-\varepsilon)}\right).$$

Proof. The proof can be found in the Appendix C. □

We end this section by *uniting* the flawless and relaxed settings, arguing that Algorithm 1 can be seen as a particular case of the SPRT used in Algorithm 2.

Proposition 1. *Given fixed $\mu \in (0,1)$ and $\varepsilon = \beta\mu$. When $\beta \to 1^-$, Algorithm 2 with any $\alpha > 0$ reduces to Algorithm 1.*

Proof. The flawless setting only allows for perfectly safe machines, discarding any arm at first sign of damage $D_t = 1$. We will show that this coincides with a Sequential Probability Ratio Test that compares $\mathcal{H}_0 : \mu_a \leq 0$ vs. $\mathcal{H}_1 : \mu_a > \mu$. This will hold for any $\mu \in (0,1)$ and for all $\alpha > 0$.

For a fixed μ, consider the test defined in (7) with $\varepsilon = \beta\mu$. As $\beta \to 1^-$, the null hypothesis $\mathcal{H}_0$ becomes $\mu_a = 0$. We show that the log-likelihood ratio $\Lambda_a(t)$ goes to infinity at first sign of damage, thus declaring $\mathcal{H}_1$ and terminating the SPRT. A sufficient statistic for computing the log-likelihood ratio $\Lambda_a(t)$ is counting the k outcomes of $D_\tau = 1$ in a total of t pulls. Then $\Lambda_a(t) = k\log\frac{\mu}{\mu-\varepsilon} + (t-k)\log\frac{1-\mu}{1-\mu+\varepsilon}$. Writing $\varepsilon = \beta\mu$, $\Lambda_a(T) = t\log\frac{1-\mu}{1-\mu(1-\beta)} + k\log\frac{1-(1-\beta)\mu}{(1-\beta)(1-\mu)} \xrightarrow[\beta\to 1^-]{} \infty\ \forall k > 0$. Then the SPRT decides on the alternative $\mathcal{H}_1$ at first sign of damage, no matter how small α is. □

III. Assured Reinforcement Learning

This section builds on the insights given by the MABs to detect and discard unsafe policies in the context of RL. In this new context, unsafe policies are those which lead to a probable constraint violation, either immediately or further along the trajectory. We focus on the *Flawless setting* ($\mu = 0$), identifying all unsafe actions with a nonzero probability of damage. We start this section by defining the RL setup, including the underlying Markov Decision Process and the constrained value maximization problem to solve. Then, we proceed to write an equivalent unconstrained problem by adding a hard-barrier index to the cost, which acts as a constraint enforcer. This equivalent formulation allows us to derive a separation principle together with our novel barrier function, which provides a certificate of safety for the behavior policy. The barrier function stands alone, satisfying a Bellman Equation that describes the set of feasible safe policies. Naturally, this Bellman Equation yields a stochastic *Barrier learner* algorithm, that uses the damage signal observed from the system to detect and reveal unsafe policies. We finish this section showing that all unsafe policies can be detected in expected finite time, and present an Assured Q-learning Algorithm that makes use of the Barrier-learner.

A. Value function decomposition

With the goal of defining our constrained RL problem, consider a Markov Decision Process $\mathcal{M}$ with finite state space $\mathcal{S}$, finite action space $\mathcal{A}$, a reward set $\mathcal{R}$, and a damage indicator belonging to $\{0,1\}$. A transition kernel p, specifies the conditional transition probability $p(s',r,d \mid s,a) := P(S_{t+1}=s', R_{t+1}=r, D_{t+1}=d \mid S_t = s, A_t = a)$, from state $s \in \mathcal{S}$ to state $s' \in \mathcal{S}$ resulting in a reward $r \in \mathcal{R}$ and a damage indicator d, all under action $a \in \mathcal{A}$. Constraints must be satisfied so that the agent does not receive damage along the trajectory, which amounts to imposing

$$\mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t D_{t+1} \;\middle|\; S_0 = s\right] \leq 0. \tag{14}$$

In this context, our goal is to maximize the value function for each possible starting state while ensuring the constraint satisfaction in the long run, i.e.,

$$V^*(s) := \max_{\pi}\mathbb{E}_\pi\left[\sum_{t=0}^{\infty}\gamma^t R_{t+1} \;\middle|\; S_0 = s\right] \tag{15}$$

$$\text{s.t.:}\quad \mathbb{E}_\pi\left[\sum_{t=0}^{\infty}\gamma^t D_{t+1} \;\middle|\; S_0 = s\right] \leq 0. \tag{16}$$

where both expectations are taken over the trajectories induced by π. While this is a case of a cumulative constraint in expectation which could be put in Lagrangian form (see e.g., [25] [37]), we take an alternative approach that will lead to finite time detection. Notice that because D_t is, by definition, non-negative, we can decouple this constraint and force each damage indicator to be null, that is,

$$\mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t D_{t+1} \;\middle|\; S_0 = s\right] \leq 0 \iff D_{t+1} = 0 \ a.s.\ \forall t. \tag{17}$$

Notice that the factor γ^t in the constraint of (16) is unsubstantial. On the other hand, its presence in the cost is retained to ensure that the sum of rewards remains bounded. However, γ could be also set to one in the cost of (16) if the number of times the system outputs a non-zero reward is finite, as it would be the case of an MDP with an almost-sure absorbent state with zero damage.

Given (17), we proceed to rewrite (15)–(16) as an equivalent problem with one constraint per time period,

$$V^*(s) := \max_\pi \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t R_{t+1} \mid S_0 = s \right] \tag{18}$$

$$\text{s.t.:} \quad D_{t+1} = 0 \quad a.s. \;\; \forall t \tag{19}$$

We chose to work with (18)–(19) as our most general constrained RL problem to solve, although we could specialized it further by shaping the transition kernel $p(s', r, d \mid s, a)$. For instance, we could define a feasible set $\mathcal{F} \subset \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ so that by setting $P(D_{t+1} = 0 \mid (S_t, A_t, S_{t+1}) \in \mathcal{F}) = P(D_{t+1} = 1 \mid (S_t, A_t, S_{t+1}) \notin \mathcal{F}) = 1$ we would model the damage indicator as being activated whenever the transition triplet (S_t, A_t, S_{t+1}) transgresses $\mathcal{F}$. This particular case could be further specialized to the case where the damage indicator is turned on whenever S_{t+1} breaks off from a feasible set $\mathcal{F}_S \subset \mathcal{S}$, for instance if a navigation agent crashes to a wall, or if the action A_t violates a feasible set $\mathcal{F}_A \subset \mathcal{A}$; e.g., when exceeding a maximum force.

Returning to our general formulation in (19), let us define the value function V^π for a specific policy π, in which the constraints are embedded inside the expectation.

$$V^\pi(s) := \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \left(\gamma^t R_{t+1} + \mathbb{I}\left[D_{t+1}\right] \right) \mid S_0 = s \right] \tag{20}$$

where the hard barrier index function $\mathbb{I}\left[\cdot\right]$ takes the form

$$\mathbb{I}\left[D_{t+1}\right] = \log\left(1 - D_{t+1}\right) = \begin{cases} 0 & \text{if } D_{t+1} = 0 \\ -\infty & \text{if } D_{t+1} = 1 \end{cases} \tag{21}$$

so that it is null when the transition is safe and takes the value $-\infty$ in the event of an unsafe transition. Being (21) unbounded, expectations are defined in the sense of the Lebesgue integral for functions in the extended real line [38].

The proposed value function definition (20) will prove useful in two senses: firstly, we will show that maximizing (20) is the same as (18)–(19). Secondly, the additional term in (20) will allow for a barrier-based decomposition of the value function, which will aid in the learning of constraints.

Lemma 4 (Equivalence)**.** *Problem* (15) *is equivalent to the maximization of* (20)*, that is*

$$\max_\pi \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \left(\gamma^t R_{t+1} + \mathbb{I}\left[D_{t+1}\right] \right) \mid S_0 = s \right] \tag{22}$$

Proof. If a policy π_0 is unfeasible for Problem (18)–(19), then $\exists t : P\left(D_{t+1} = 1\right) > 0$. This non-zero probability renders the expected value in (22) to $-\infty$ for π_0. Conversely, if a policy π_1 is feasible for (22) then it must necessarily hold that $D_{t+1} = 0$ almost surely $\forall t$, and hence π_1 is feasible for (18)–(19) as well. Therefore the feasible sets of both problems coincide. Lastly, for every feasible policy it must hold $\log\left(1 - D_{t+1}\right) = 0 \; \forall t$, in which case the function being maximized is the same. Then the optimal sets of the two problems coincide. □

While solving (18)–(19) is of our utmost interest, we have just shown that, to this end, we can solve (22) instead. In what follows we will take this one step further, and show that (20) admits a *barrier-based decomposition* and can be cast as the sum of two value functions: one that checks only whether the policy in consideration is feasible (which will be the main focus of this work) and one that optimizes the return, provided the policy is feasible. The main idea behind this decoupling being that the search for feasible policies will be, in practice, an easier task to undergo.

To this end we define an auxiliary *hard-barrier* value function H^π that will relate to V^π.

$$H^\pi(s) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \log\left(1 - D_{t+1}\right) \mid S_0 = s \right] \tag{23}$$

We proceed similarly for the action-value function Q^π and its barrier counterpart B^π

$$Q^\pi(s,a) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \left(\gamma^t R_{t+1} + \mathbb{I}\left[D_{t+1}\right] \right) \middle| S_0 = s, A_0 = a \right]$$

$$B^\pi(s,a) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \log\left(1 - D_{t+1}\right) \middle| S_0 = s, A_0 = a \right] \tag{24}$$

Searching for policies that are optimal for (22) for each possible state is our original goal. By contrast, a problem such as maximizing (23) is one that seeks to find *safe* policies, in the sense that they are *feasible* for (22). The main idea underpinning our work is that we can jointly work on optimizing (23), which reduces the search over the policy space, while at the same time maximizing the return present in (22). In the following Theorem we establish a fundamental decomposition relationship between the value functions and their auxiliary counterparts.

Theorem 5 (Barrier-based decomposition)**.** *Assume rewards R_{t+1} are bounded almost surely for all t. Then, for every policy π*

$$V^\pi(s) = V^\pi(s) + H^\pi(s) \tag{25}$$

$$Q^\pi(s,a) = Q^\pi(s,a) + B^\pi(s,a) \tag{26}$$

Proof. We shall prove (25) only, since the proof for (26) is alike. The following identities hold, as explained below.

$$V^\pi(s) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \left(\gamma^t R_{t+1} + \log\left(1 - D_{t+1}\right) \right) \mid S_0 = s \right]$$

$$= \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \left(\gamma^t R_{t+1} + \log\left(1 - D_{t+1}\right) \right) \mid S_0 = s \right] \tag{27}$$

$$+ \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \log\left(1 - D_{t+1}\right) \mid S_0 = s \right] \tag{28}$$

To show that $V^\pi(s)$ can be separated in (27) and (28), first suppose policy π is feasible (for Problem (22)). This necessarily implies that $D_{t+1} = 0 \;\; a.s. \;\; \forall t$, which makes the second term in (27) vanish. Conversely, suppose that the policy in consideration is infeasible. This together with the fact that rewards are bounded almost surely yields $V^\pi(s) = -\infty$, which is the same value attained by both (27) and (28). □

The preceding result implies non-trivial consequences. If the learning agent can interact with the environment and have access to rewards R_{t+1} and queries of whether a transition has been safe (i.e. $D_{t+1} = 0$), then it can separately learn both $Q^{\pi}(s,a)$ and $B^{\pi}(s,a)$. This is discussed in the next remark.

Remark 4 (Properties of the optimal barrier function B^{*}).

$$\begin{aligned} B^{*}(s,a) &= \max_{\pi} B^{\pi}(s,a) \\ &= \max_{\pi} \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \log\left(1 - D_{t+1}\right) \;\middle|\; S_0 = s, A_0 = a \right] \end{aligned} \tag{29}$$

By definition, the entries of $B^{\pi}(s,a)$ will either be 0 or $-\infty$ for any policy π. Having $B^{*}(s,a) = -\infty$ means that if starting at s with action a and then following any policy, there is an (albeit small) non-zero probability that an unsafe event will be encountered. Conversely, if $B^{*}(s,a) = 0$ then following the optimal policy guarantees we will always ensure constraints while starting from state s and action a. Furthermore, if B^{*} is available, then any policy that chooses an action such that $B^{*}(s,a) = 0$ is a feasible policy for that state. If, on the other hand, the policy chooses an action which leads to $B^{*}(s,a) = -\infty$, then that policy can be deemed unsafe immediately, and can be discarded. Notice, however, that this is a particular property of the optimal function B^{*}: for a sub-optimal π, B^{π} can be padded with lots of $-\infty$ in places where B^{*} has zeros. Intuitively, this would mean that the sub-optimal policy starting from (s,a) makes unsafe transitions along its trajectory. In this sense, having access to the optimal action-value function B^{*} helps constraint the search of any other known algorithms (such as Q-Learning) to only feasible policies. This becomes a joint work of optimizing the return (using one of many possible techniques), while learning the feasible set of actions at the same time.

B. Barrier learning

Given the previous discussion on the properties of the Barrier function $B(s,a)$, we will focus on the feasibility problem aiming to learn the optimal $B^{\star}(s,a)$ from data. Before that, we state the following optimality condition, in the standard form of the Bellman's equations.

Theorem 6 (Bellman equation for $B^{\star}(s,a)$). *The optimal barrier function satisfies*

$$B^{\star}(s,a) = \mathbb{E}\left[\mathbb{I}\left[D_{t+1}\right] + \max_{a' \in \mathcal{A}} B^{\star}(s',a') \;\middle|\; S_t = s, A_t = a \right]. \tag{30}$$

Proof. It follows from Proposition 4.1.1 in [39, pp. 217] with the value function being *minimized*, and assuming possibly unbounded but non-negative costs $C(S_t, A_t) = -\mathbb{I}[D_{t+1}]$. $\square$

Besides providing a certificate for optimality, the Bellman Equations for $B^{\star}(s,a)$ hint towards a stochastic iterative algorithm to optimize (29) from data, the same way the Q-learning algorithm is derived from the standard unconstrained value function [39]. We will elaborate on this stochastic algorithm next. As introduced in (29), our goal is to learn a *safe* policy π with an associated optimal Barrier function $B^{\star}(s,a)$ that encodes the trajectories that satisfy the constraints at all time steps almost surely. For this purpose we first devise an iterative algorithm that attempts to reach a fixed point satisfying (30), that would be

$$B_{k+1}(s,a) = \mathbb{E}\left[\mathbb{I}\left[D_{t+1}\right] + \max_{a' \in \mathcal{A}} B_k(s',a') \;\middle|\; S_t = s, A_t = a \right]. \tag{31}$$

Next, we appeal to the stochastic approximation machinery [40] to drop the unknown expectations yielding a data driven version of (31). The resulting stochastic update is given next.

```
Algorithm 3: barrier_update
  Input: $B$-function and $(s_t, a_t, s_{t+1}, d_{t+1})$ tuple
  Output: Barrier-function $B$
    $B(s_t,a_t) \leftarrow B(s_t,a_t) + \log(1-d_{t+1}) + \max_{a'} B(s_{t+1},a')$
    return $B$
```

The update in Algorithm 3 incorporates the information carried in d_{t+1}, which signals whether the constraint has been violated or not during the transition from time t to $t+1$. Moreover, the update does not only consider immediate violations, but also the future effect of the action a_t that is summarized in the second term $\max_{a'} B(s_{t+1},a')$. This *bootstrapping* mechanism leverages on stationarity to collect damage information from all previous state transitions, and summarize it in $B(s_{t+1},a')$ which predicts long-run future effect of the state-action pairs at time $t+1$. Thus, by repeating the update in Algorithm 3 with new data coming from successive system interactions, an agent can synthesize the whole information about all past constraint violations in the barrier function $B(s,a)$ for unveiling the set of unsafe policies.

We leave the details of this iterative algorithm and its performance guarantees for the next subsection. Now that we have introduced this data-driven strategy for securing the environment, let us turn back our attention to the implications of our decomposition result in Theorem 5. The identity (26) can be used to embed the safety information provided by $B(s,a)$ in the Q-function $Q(s,a)$. Hence, we identify that the condition $Q(s,a) = -\infty$ is equivalent to $B(s,a) = -\infty$ and this propagates information about safety from successor states. For those which do satisfy the constraints we have $B(s,a) = 0$, and then $Q(s,a)$ will carry the information of the observed rewards. The following update is complementary to Algorithm 5, and amounts to the standard Q-learning algorithm for maximizing rewards at safe pairs of states and actions.

Next we specify how to use Algorithm 3 to ensure safety, and demonstrate the sample complexity of the learning process. After that, we combine algorithms 3 and 4 with the goal of safely maximizing rewards. We will borrow the well-known convergence results of Q-learning [41] together

Algorithm 4: `q_update`

```
Data: Step size $\eta$, discount factor $\gamma$
Input: Functions $Q$, $B$, and $(s_t, a_t, s_{t+1}, r_{t+1})$
Output: Q-function
    $Q(s_t,a_t) \leftarrow (1-\eta)Q(s_t,a_t)+\eta\Big(r_{t+1}+\gamma \max_{a'} Q(s_{t+1},a')\Big)$
    $Q(s_t,a_t) \leftarrow B(s_t,a_t)+Q(s_t,a_t)$
return $Q$
```

with our separation principle in Theorem 5 to provide performance guarantees for our novel Assured Q-learning algorithm.

C. Performance analysis of Barrier-Learner

First, we consider a simple barrier learner algorithm where one can query on any state-action pair (s, a) and obtain sampled transition (s, a, s', d) according to $P\left(S_1 = s', D_1 = d | S_0 = s, A_0 = a\right)$. The barrier learner is shown in Algorithm 5. Our analysis shows that the expected queries/samples required until all unsafe state-action pairs are detected is finite, given that any non-zero transition probability is lower bounded by μ.

Algorithm 5: Barrier Learner Algorithm

```
Data: Constrained Markov Decision Process $\mathcal{M}$
Result: Optimal action-value function $B^*$
Initialize $B^{(0)}(s,a) = 0, \forall (s,a) \in \mathcal{S} \times \mathcal{A}$
for $t = 0, 1, \cdots$ do
    Draw
      $(s_t, a_t) \sim \mathrm{Unif}(\{(s,a) : B^{(t)}(s,a) \neq -\infty\})$
    Sample transition $(s_t, a_t, s'_t, d_t)$ according to
      $P\left(S_1 = s'_t, D_1 = d_t | S_0 = s_t, A_0 = a_t\right)$
    $B^{(t+1)} \leftarrow$
      barrier_update($B^{(t)}, s_t, a_t, s'_t, d_t$)
end
```

In an MDP, an (s, a) pair is unsafe ($B^*(s, a) = -\infty$) if either it immediately causes damage, i.e. $P(D_1 = d|S_0 = s, A_0 = a) > 0$, or it transitions to an unsafe state, namely $P(S_1 = s'|S_0 = s, A_0 = a) > 0$ for some s' with $B^*(s', a) = -\infty, \forall a \in \mathcal{A}$. As a result, when an (s, a) is taken, one may observe the damage after several steps. We let L be the *lag* of the MDP, which is the maximum steps one need to wait until observing the potential damage by taking an unsafe (s, a) pair. The exact definition of L is given in Appendix D. Regarding Algorithm 5, we have the following Theorem.

Theorem 7. *Given an MDP. Assume that exists $\mu > 0$ such that the transition probability $P(S_1 = s'|S_0 = s, A_0 = a)$, is either zero or lower bounded by μ, for all $s, s' \in \mathcal{S}, a \in \mathcal{A}$. Let T be earliest time when Algorithm 5 detects all unsafe state-action pairs, i.e. $T := \min\{t : B^{(t)} = B^*\}$, then we have*

$$\mathbb{E}[T] \leq (L+1)\frac{|\mathcal{S}||\mathcal{A}|}{\mu}\left(\sum_{k=1}^{|\mathcal{S}||\mathcal{A}|}\frac{1}{k}\right), \tag{32}$$

where L is the lag of the MDP.

Proof Sketch. Theorem 7 is proved in three steps. We refer the readers to Appendix D for the complete proof.

First, we reformulate the algorithm so that the sampling process of (s_t, a_t) is independent of the current progress of the $B^{(t)}$-function: at iteration t, one samples an (s_t, a_t) pair uniformly from the entire $\mathcal{S} \times \mathcal{A}$ set, then the algorithm chooses to either accept or reject the sample depending on the value of $B^{(t)}(s_t, a_t)$. This way, we turn to study the number of acceptance by the reformulated algorithm before it detects all unsafe state-action pairs. Secondly, we provide a modified algorithm with more restrictions on accepting the sample compared to the reformulated algorithm. Such restrictions allow the modified algorithm to learn unsafe state-action pairs in multiple stages: at each stage, the algorithm only allows to detect a subset of unsafe state-action pairs, but the detection process in every stage can be viewed as a safe multi-arm bandits problem discussed in Section II.

Lastly, we derive an upper bound on the expected number queries of the modified algorithm until successfully detecting all unsafe state-action pairs, based on Theorem 2, which is also an upper bound on $\mathbb{E}[T]$.

□

While the bound for $\mathbb{E}[T]$ in Theorem 7 is admittedly loose (we use a lower-bound μ for the transition probabilities and a provably slower surrogate algorithm with more restrictions), it serves the purpose of upholding our main claim in the paper, which is that all unsafe policies can be detected in finite time. The resulting simplicity of the bound in (32), also let us observe the fundamental factors adding to the detection time T. Specifically, with larger spaces $\mathcal{S}$ and $\mathcal{A}$ more exploration is needed, with a smaller μ or longer lag L unsafe actions take longer to be revealed as damage, all three factors adding to a longer detection time. A tighter bound is presented in Appendix D when we prove Theorem 7.

The Barrier Learning Algorithm 5 stands alone as a data-driven method to assess safety feasibility. However, our separation principle allow us to combine it with standard existing reward optimization algorithms in order to add safety. For instance, by wrapping Algorithm 5 around the acclaimed Q-learning algorithm we obtain a Generative Assured Q-learning method to maximize the rewards over the set of safe policies, as is shown in Algorithm 6.

As a corollary of Theorem 7, and borrowing the convergence results of Q-learning from [41] we establish the following result.

Corollary 1. *With finite state space $\mathcal{S}$ and action space $\mathcal{A}$, bounded rewards $R_t \leq C$, and diminishing step-sizes satisfying $\sum_t \eta_t = \infty$ and $\sum_t \eta_t^2 < \infty$, the iterates $Q^{(t)}$ of the Algorithm 6 converge almost surely to the optimal Q-function $Q^\star$ satisfying*

$$\lim_{t\to\infty} Q^{(t)}(s,a) = Q^*(s,a) \ (w.p.1) \ \ \forall (s,a) \in \mathcal{S}\times\mathcal{A}$$

with

$$Q^*(s,a) = \mathbb{E}\Big[\log\left(1 - D_{t+1}\right)$$

Algorithm 6: Generative Assured Q-Learning

```
**Data:** Constrained Markov Decision Process $\mathcal{M}$
**Result:** Optimal action-value functions $B^*$ and $Q^*$
Initialize
  $B^{(0)}(s,a) = 0,\ Q^{(0)}(s,a) = 0 \forall (s,a) \in \mathcal{S} \times \mathcal{A}$
**for** $t = 0, 1, \cdots$ **do**
    Draw $(s_t, a_t) \sim \text{Unif}(\{(s,a) : \mathcal{B}(s,a) \neq -\infty\})$
    Sample transition $(s_t, a_t, s_t', d_t)$ according to
      $P(S_1 = s_t', D_1 = d_t | S_0 = s_t, A_0 = a_t)$
    $B^{(t+1)} \leftarrow$
      barrier_update$(B^{(t)}, s_t, a_t, s_t', d_t)$
    $Q^{(t+1)} \leftarrow$ q_update$(B^{(t)}, Q^{(t)}, s_t, a_t, s_t', r_t)$
**end**
```

$$+ \max_{a' \in \mathcal{A}} Q^*(s', a') \;\Big|\; S_t = s, A_t = a \Big]. \quad (33)$$

Moreover, all unsafe state action pairs corresponding to $Q^*(s,a) = -\infty$ *are detected in expected finite time* $T_{s,a}$ *such that*

$$Q^{(t)}(s,a) = -\infty, \ \forall\ t \geq T_{s,a},$$

and

$$\mathbb{E}\left[T_{s,a}\right] \leq \frac{|\mathcal{S}|^2|\mathcal{A}|}{\mu} \left(\sum_{k=1}^{|\mathcal{S}||\mathcal{A}|} \frac{1}{k} \right). \quad (34)$$

Proof. In order to apply the convergence results of [41] we need iterates $Q^{(t)}$ to be finite for all t. But this is guaranteed by Theorem 7 for all safe pairs (s,a) such that $B^*(s,a) = 0$. Indeed, or all values such that $B^*(s,a) = 0$, we have $B^{(t)} = 0$ for all t and the updates of the function $Q^{(t)}$ in Algorithm 6 amount to the asynchronous Q-learning updates with finite rewards and diminishing step-size required by [41].

For the pairs such that $B^*(s,a) = -\infty$, Theorem 5 implies $Q^*(s,a) = -\infty$ and according to Theorem 7 there must exist a time instant $T_{s,a}$ satisfying (34) such that $B^{(t)} = -\infty \ \ \forall t \geq T_{s,a}$. Since by construction the q_update in Algorithms 6 and 3 implies that $Q^{(t)} = -\infty$ whenever $B^{(t)} = -\infty$, then $\lim_{t \to \infty} Q^{(t)}(s,a) = -\infty$ for all pairs (s,a) such that $B^*(s,a) = Q^*(s,a) = -\infty$. $\square$

The previous result applies to the specific case of diminishing step-sizes and immediate restarts after episodes of length one. However, Q-learning is widely applied with longer episodes and convergence is guaranteed provided that each state-action pair is visited infinitely often While one step episodes in Algorithm 6 were used in order to simplify the proof of Corollary 1, the numerical experiments of the next section will demonstrate that these safe convergence results carry out to an episodic version of Assured Q-learning with ϵ-greedy exploratory modes.

IV. Numerical Experiments

We now proceed to some numerical experiments that back up the results presented throughout the paper. We first focus on the multi-armed bandit setup and on the problem of detecting unsafe machines under a uniform strategy, then we demonstrate the barrier learner and the characterization of a globally unsafe MDP and how this can be achieved in finite time, and finally we compare our Assured Q-learning algorithm against classic Q-learning on a grid-world.

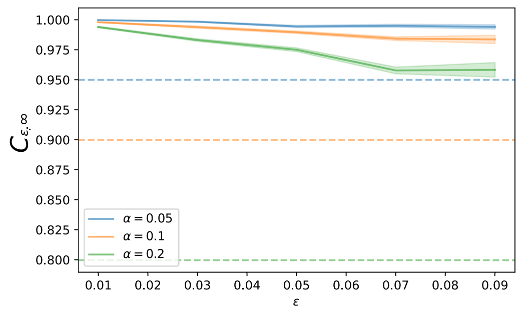

Fig. 1. Final safety ratio $C_{\varepsilon,\infty}$ for $\mu = 0.1$ as a function of ϵ, for varying α. More safe machines are kept when using small values of ϵ and α, but this in turn implies longer training time (c.f. Figure 2)

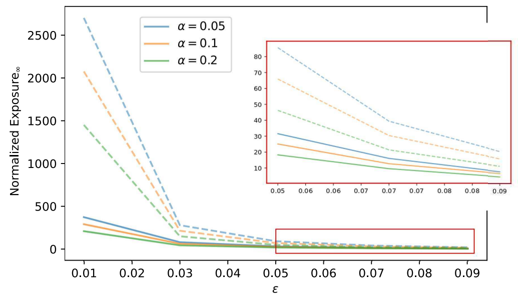

Fig. 2. Normalized final exposure for $\mu = 0.1$ as a function of ϵ, for varying tolerance level α. Larger values of α and ϵ achieve lower handicap (which implies faster detection).

A. Multi-armed bandits

We illustrate the performance of the Relaxed Inspector (Algorithm 2) on a setup of $K = 1000$ arms, with a safety requirement of $\mu = \frac{1}{10}$. Each arm's true safety parameter is uniformly sampled between 0 and $\frac{1}{5}$. This means that if $\epsilon \approx 0$, around half of the machines are safe and the other half unsafe. We run the Relaxed Inspector on this setting under a uniform strategy, for varying levels of both the failure tolerance α and the slack ϵ. After all unsafe machines have been detected —which happens and is certified to be done in finite time in virtue of Theorem 3— we consider $C_{\varepsilon,\infty}$, the final conservation ratio and the normalized final exposure $\frac{1}{K}E_\infty$. Figures 1 and 2 show both this metrics for varying α and ϵ. The curves in these figures show the average value obtained after 16 independent runs. Shaded intervals correspond to $\pm\sigma/\sqrt{16}$, with σ being the sample deviation.

These figures certify the intrinsic trade-off in our methodology: if a large value of α is used, one can obtain low exposure E_t (less pulls of unsafe machines), but at the cost of discarding more safe machines (smaller conservation ratio $C_{\varepsilon,t}$). For further examples we refer the reader to [1].

B. *Barrier learner*

We now turn to identification of unsafe states on MDPs via the barrier learner algorithm. We will try our methodology in the *unstable grid-world* shown on the upper-left panel of Fig. 3. At each time step, the agent starts in one of the white states and takes an action which is up, down, left or right. Moving between white states is allowed. Bumping into walls —black states— is ok as well, and leaves the agent at the same place he was in. Purple states are *holes*, and moving into a hole is *unsafe* ($D_t = 1$) Whenever an action is taken, there is a probability $p = 0.6$ that the agent moves to the corresponding neighboring state. However, with probability $1-p = 0.4$ he moves to a random neighboring state instead. Given this *unstable* nature of the state transitioning, it is clear that states adjacent to the holes $s_{\texttt{adj}}$ are unsafe. Indeed, the probability to be damaged at $s_{\texttt{adj}}$ is at least $\frac{1-p}{4} = 0.1$. Notice though that the whole state-action space is unsafe, since with every action there is a probability of transitioning to a random neighboring state. The question then is how long it takes the barrier learner to characterize the whole state space as unsafe.

The top section of Fig. 3 shows the evolution of the barrier learner algorithm on this grid. At each step the starting state s_0 is taken uniformly, and then a random action is sampled. When the agent is assured that $B(s_0, a) = -\infty \ \forall a$ it marks that state as unsafe, and paints it red in Fig. 3. At first, only the states close to the holes are classified as unsafe. Soon enough, detection of unsafe states stems from the holes to the edges of the grid. Near the $t = 10000$ mark all states are correctly detected, as is shown in the bottom plot of Fig. 3. There we show the proportion of unsafe states detected by the barrier learner, along with the bound obtained in Theorem 7, where we use $\sum_{k=1}^{|\mathcal{S}||\mathcal{A}|} \frac{1}{k} \sim \log\left(|\mathcal{S}||\mathcal{A}|\right)$. In this case the non-zero transition probabilities of the MDP are lower-bounded by $\mu = \frac{1-p}{4} = 0.1$. The bound is admittedly loose as we discussed in Section II, but demonstrates our main claim in 7 about being able to detect unsafe actions in finite time.

C. *Towards learning safely*

In this part we compare the performance of Classic Q-learning against a modified version which makes use of the barrier learner, which we dub Episodic Assured Q-learning. This algorithm works in the same fashion as Algorithm 6, but it runs episodes following a behavioral policy instead of drawing state-action pairs generatively.

Let us first describe the environment. Consider an agent that is bound to navigate a narrow corridor, as shown in Fig. 4. Each episode starts on the leftmost state and the goal is to reach the rightmost state. The agent can move deterministically between adjacent states by taking actions up-down-left-right. However, bumping into a wall is considered unsafe ($D_t = 1$) and terminates the episode. For a fair comparison, for the classic Q-learning setup we let the reward be $-\infty$ whenever the agent bumps into a wall. In both settings the agents follow a behavioral policy that is ϵ-greedy with respect to the policy derived by their learnt Q-function, that is to say, selects $a = \operatorname{argmax}_{a'} Q(s, a')$ with probability

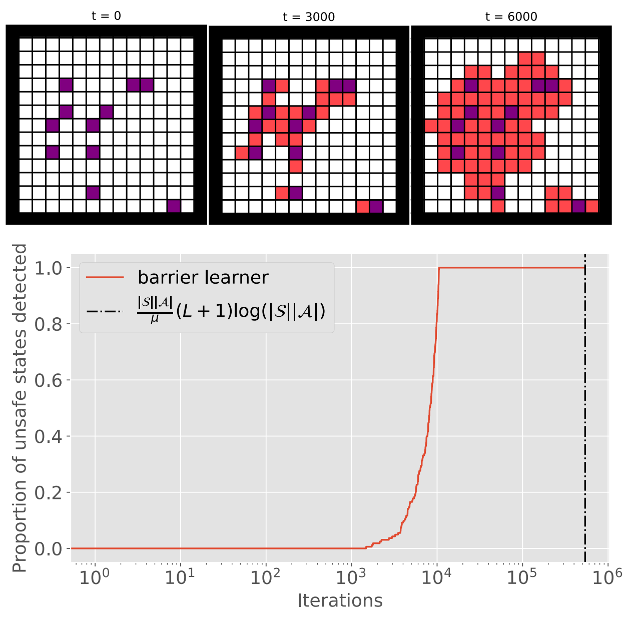

Fig. 3. On the top: evolution of the Barrier Learner Algorithm on a 15 by 15 unstable grid. As time advances, detection of unsafe states (in red) backpropagates from the immediately unsafe holes (purple), eventually classifying the whole state space as unsafe. On the bottom: proportion of states classified as unsafe as a function of time. All states are correctly identified in finite time, respecting the bound of Theorem 7.

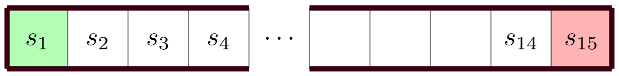

Fig. 4. Example grid-world: narrow corridor of length 15. An agent starts each episode at the green state and must learn to reach the red state. Actions are up-down-left-right, and they move the agent to the adjacent state deterministically, if possible. Bumping into a purple wall is *unsafe*, and leads to episode termination. Transitioning to the red state has reward +100. All other rewards are zero.

$1-\epsilon$ or takes a uniformly random action with probability ϵ. We fix $\epsilon = 0.1$ and the learning-rate $\alpha = 0.1$.

As a metric of performance, we count the number of steps it takes for the agent to reach the goal for the first time, which is shown in Fig. 5. Reaching the goal is not as straightforward as it seems, since whenever the agent bumps into a wall he gets sent to the leftmost state. A histogram depicting the number of wall bumps before reaching the goal for the first time is show in Fig. 6. Notice that this is exactly the same as counting the number of episodes taken to reach the goal.

As is evident of both figures, the assured agents generally outperform their classic counterparts. In particular, training time for the assured agents is faster —see e.g. Fig. 5, where the total number of transitions for these agents has both smaller sample mean and sample variance. The reason for this performance improvement is the fact that the assured agents quickly *learn* that there are actions that must not be taken again, while standard ϵ-greedy Q-learning keeps drawing poor actions indefinitely.

V. Conclusions

In this work we address the problem of learning to act safely in unknown environments. We make the case that learning safe policies is fundamentally different from

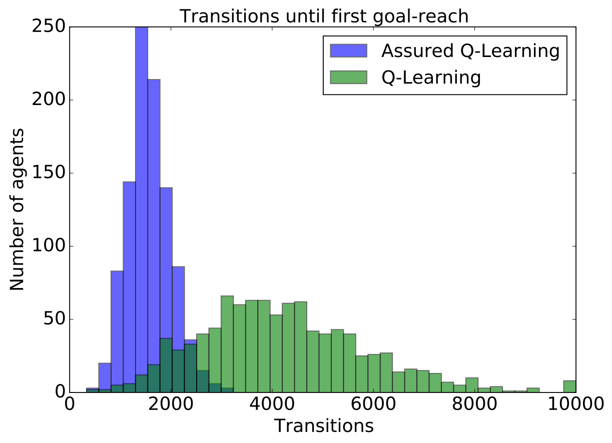

Fig. 5. Comparison between Assured Q-learning (blue) and classic Q-learning (green) on the time taken to reach the goal (s_{15}) on the narrow corridor of Fig. 4. To build the histogram we train 1000 agents independently with each algorithm.

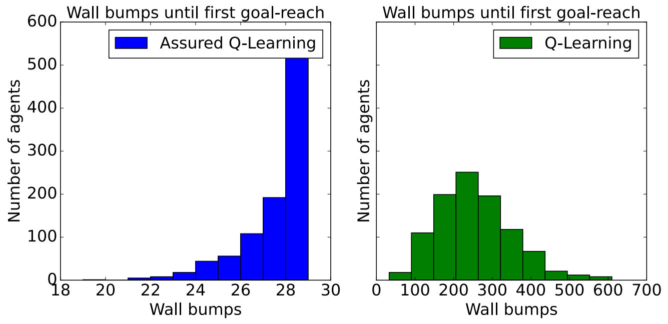

Fig. 6. Comparison between Assured Q-learning (blue) and classic Q-learning (green) on the number of bumps into walls (i.e. episodes) until first reaching the goal (s_{15}) on the narrow corridor of Fig. 4. To build the histograms we train 1000 agents independently with each algorithm. Mind the difference on the x-axis between graphs.

learning optimal policies, and is a task that can be done separately and in a more efficient manner. By incorporating in the model a binary *damage signal* that indicates constraint violations, our results show that we can identify all unsafe actions (MABs) and state-action pairs (MDPs) in expected finite time with probability one guarantees. These results imply that the learner is not indefinitely exposed to damage, and could aid in the design of new algorithms that rapidly learn to act safely while jointly optimizing returns.

Appendix

A. Proof of Theorem 2

We prove Theorem 2 by induction.

Proof. Each iteration of Algorithm 1 can be view as doing a Bernoulli trial with success rate being the probability of detecting an unsafe machine.

When $M = 1$, the algorithm repeats the same Bernoulli trial until a success, the success rate for that Bernoulli trial is

$$P(\text{“detecting an unsafe machine”})$$
$$= P(\text{“the unsafe machine is selected and detected”}) = \frac{\mu_1}{K},$$

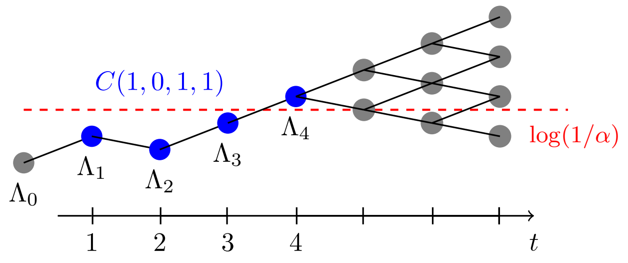

Fig. 7. Visualization of the cylinder set $C(1,0,1,1)$. It contains all trajectories $\{x_\tau\}_{\tau=1,\ldots,\infty}$ that start with $1,0,1,1$.

Then T is a geometric random variable with parameter $\frac{\mu_1}{K}$, hence $\mathbb{E}T = \frac{K}{\mu_1} \le \frac{K}{\mu_{\text{low}}}$.

Now suppose Theorem 2 holds for $M = m$, consider detecting $m+1$ unsafe machines among in total K machines, the algorithm repeat the same Bernoulli trial until one success, the success rate for that Bernoulli trial is

$$P(\text{“detecting an unsafe machine”})$$
$$= \sum_{k=1}^{m+1} P(\text{“unsafe machine } k \text{ is selected and detected”})$$
$$= \sum_{k=1}^{m+1} \frac{1}{K} \cdot \mu_k \,,$$

Let $T_1 := \min\{t : D_t = 1\}$ be the time epoch for the first success. Then T_1 is a geometric random variable with parameter $\sum_{k=1}^{m+1} \frac{\mu_k}{K}$, hence

$$\mathbb{E}[T_1] = \left(\sum_{k=1}^{m+1} \frac{\mu_k}{K} \right)^{-1} \le \left(\sum_{k=1}^{m+1} \frac{\mu_{\text{low}}}{K} \right)^{-1} = \frac{K}{\mu_{\text{low}}} \frac{1}{M+1} \,. \tag{35}$$

Now $T - T_1$ is exactly distributed as the time Algorithm 1 takes to detect m unsafe machines among in total $K-1$ machines. By our induction assumption, we have

$$\mathbb{E}[T - T_1] \le \frac{K-1}{\mu_{\text{low}}} \left(\sum_{k=1}^{m} \frac{1}{k} \right) \le \frac{K}{\mu_{\text{low}}} \left(\sum_{k=1}^{m} \frac{1}{k} \right) . \tag{36}$$

Adding (35) and (36), we obtain the desired result

$$\mathbb{E}[T] \le \frac{K}{\mu_{\text{low}}} \left(\sum_{k=1}^{m+1} \frac{1}{k} \right) ,$$

for the case $M = m + 1$, which finishes the proof. □

B. Proof of Lemmas 2–3

In order to prove these lemmas we first review cylinder sets, which were first described by Wald [35]. Consider an infinite sequence $\{d_\tau\}, \tau \ge 1$ and define C_∞ as the space of all such sequences. The set $C(x_1, \ldots, x_t)$ is called a cylinder set of order t, and is defined as the subset of C_∞ which collects sequences with $d_1 = x_1, \ldots, d_t = x_t$. A cylinder set will be said to be of the *unsafe type* if

$$\Lambda_t = \log \frac{f_\mu (x_1, \ldots, x_t)}{f_{\mu-\epsilon} (x_1, \ldots, x_t)} \ge \log(1/\alpha) \,, \tag{37}$$

and

$$\Lambda_\tau = \frac{f_\mu(x_1,\ldots,x_\tau)}{f_{\mu-\epsilon}(x_1\ldots,x_\tau)} < \log(1/\alpha) \quad \forall \tau < t\,. \tag{38}$$

The first condition guarantees that all sequences $\{d_\tau\}$ belonging to the unsafe cylinder $C(x_1,\ldots,x_t)$ will lead to the acceptance of $\mathcal{H}_1$. This will occur at time t, regardless of the future samples $d_\tau, \tau > t$. The second condition states that the threshold is not surpassed sooner than t. Conditions (37)–(38) imply that cylinders of different orders are disjoint sets, since Λ_t exceeds the threshold for the first time at t and no sooner than that, and this cannot be true for different values of t.

The union of all unsafe cylinder sets (of any order) defines the set of sequences that lead to deciding H_1. Let us name this (disjoint) union as Q_U. Let us also define Q_S as the complement of Q_U

$$Q_S = Q_U^{\complement}\,. \tag{39}$$

Q_S is the set of all sequences for which $\Lambda_t < \log(1/\alpha)$ for all t. Since the sets are complementary, it holds that

$$P_{\mu_a}(Q_U + Q_S) = 1 \quad \forall \mu_a \in [0,1]\,, \tag{40}$$

with $P_{\mu_a}(Q)$ being the probability of any subset Q of C_∞, under the assumption that the sequence is generated by i.i.d. Bernoulli random variables with parameter μ_a.

We now turn to the proof of Lemmas 2–3 by restating them in terms of the cylinder sets just described.

Lemma 2 (Restated)**.** *Let $P_{\mu_a}(Q)$ be the probability of $Q \subset C_\infty$, under the assumption that the sequence of data is generated by i.i.d. Bernoulli random variables of parameter μ_a. Then the following statements hold:*

i) Under $\mathcal{H}_1$ ($\mu_a > \mu$), i.i.d. sequences produced by such a distribution are correctly classified almost surely, that is

$$P_{\mu_a}(Q_U) = 1\,. \tag{41}$$

ii) Under $\mathcal{H}_0$ ($\mu_a \le \mu$), the probability that a trajectory never rises above $\log(1/\alpha)$ *is greater than $1-\alpha$, that is*

$$P_{\mu_a}(Q_S) \ge 1-\alpha\,.$$

Proof. We start by showing *i)*.
For a given sequence $(d_1,\ldots,d_t,\cdots)$ we have

$$\Lambda_t = \log \prod_{i=1}^{t} \frac{f_\mu(d_i)}{f_{\mu-\epsilon}(d_i)} = \sum_{i=1}^{t} \log \frac{f_\mu(x_i)}{f_{\mu-\epsilon}(x_i)}\,. \tag{42}$$

Dividing by t:

$$\frac{\Lambda_t}{t} = \frac{1}{t}\sum_{i=1}^{t} \log \frac{f_\mu(x_i)}{f_{\mu-\epsilon}(x_i)}\,. \tag{43}$$

By the Law of large numbers, as t grows (43) converges to the expectation of the right hand side under μ_a:

$$\begin{aligned}\frac{\Lambda_t}{t} &\to \mathbb{E}_{\mu_a}\left[\log \frac{f_\mu(x)}{f_{\mu-\epsilon}(x)}\right] \\ &= \mu_a\left(\log\frac{\mu}{\mu-\epsilon} - \log\frac{1-\mu}{1-(\mu-\epsilon)}\right) + \log\frac{1-\mu}{1-(\mu-\epsilon)} \\ &> \mu\left(\log\frac{\mu}{\mu-\epsilon} - \log\frac{1-\mu}{1-(\mu-\epsilon)}\right) + \log\frac{1-\mu}{1-(\mu-\epsilon)} \\ &= \mathrm{kl}(\mu,\mu-\epsilon) > 0\,,\end{aligned} \tag{44}$$

where kl is the Kullback-Leibler divergence between two Bernoulli distributions of parameters μ and $\mu-\epsilon$. The inequality holds because $\mu_a > \mu$ and the expression in brackets is positive. It follows that

$$\lim_{t\to\infty} \Lambda_t = \infty, \quad a.s.$$

Therefore there must exist a positive integer t for which Λ_t exceeds $\log(1/\alpha)$, so that the sequence $\{d_\tau\}$ belongs to an unsafe cylinder of order t and thus $\{d_\tau\} \in Q_U$, which is what we wanted to show. □

Next, we prove *ii)*. First, by showing that $P_{\mu-\epsilon}(Q_S) \ge 1-\alpha$ for the limiting case $\mu_a = \mu-\epsilon$, and then by generalizing it for $\mu_a \le \mu-\epsilon$. For $\mu_a = \mu-\epsilon$, the core of the proof relies in showing that when the threshold for declaring $\mathcal{H}_1$ is set to $\log(1/\alpha)$, then

$$P_\mu(Q_U) \ge \frac{1}{\alpha} P_{\mu-\epsilon}(Q_U)\,. \tag{45}$$

Once we prove (45), we use the fact that $P_\mu(Q_U) = 1$ (see (41)) and obtain

$$P_{\mu-\epsilon}(Q_U) \le \alpha \Rightarrow P_{\mu-\epsilon}(Q_S) \ge 1-\alpha\,, \tag{46}$$

as desired. To prove (45) we start by decomposing Q_U as the union across time t of the union of all unsafe cylinders of order t, that is

$$Q_U = \bigcup_{t=1}^{\infty} \bigcup_{(x_1,\ldots,x_t)\in\mathcal{X}_t} C(x_1,\ldots,x_t)\,,$$

where $\mathcal{X}_t$ collects the tuples $(x_1,\ldots,x_t)$ that define unsafe cylinders of order t, i.e., those satisfying (37) and (38). By construction all the cylinder sets are disjoint, hence

$$\begin{aligned}P_\mu(Q_U) &= \sum_{t=1}^{\infty} \sum_{(x_1,\ldots,x_t)\in\mathcal{X}_t} P_\mu\left(C(x_1,\ldots,x_t)\right) \\ &= \sum_{t=1}^{\infty} \sum_{(x_1,\ldots,x_t)\in\mathcal{X}_t} f_\mu(x_1,\ldots,x_t) \\ &\ge \sum_{t=1}^{\infty} \sum_{(x_1,\ldots,x_t)\in\mathcal{X}_t} \frac{1}{\alpha} f_{\mu-\epsilon}(x_1,\ldots,x_t) \\ &= \frac{1}{\alpha}\sum_{t=1}^{\infty} \sum_{(x_1,\ldots,x_t)\in\mathcal{X}_t} P_{\mu-\epsilon}(C(x_1,\ldots,x_t)) \\ &= \frac{1}{\alpha} P_{\mu-\epsilon}(Q_U)\,,\end{aligned}$$

where the second identity follows from marginalizing over future trajectories (see Fig. 7), and the inequality holds since $\mathcal{X}_t$ is defined so as to satisfy (37).

Now that we have (45), (46) follows immediately, and we move to the second part of the proof. We need to prove that $P_{\mu_a}(Q_S) \ge P_{\mu-\epsilon}(Q_S)$, or equivalently $P_{\mu_a}(Q_U) \le P_{\mu-\epsilon}(Q_U)$, when $\mu_a < \mu-\epsilon$. This essentially means that the probability of (incorrectly) classifying a safe machine as unsafe decreases as μ_a lowers, which is intuitively true.

To show this, notice that the log-likelihood ratio Λ_t can be put in terms of k, the number of outcomes of $D_\tau = 1$ over t total pulls:

$$\Lambda_t = k \log \frac{\mu}{\mu - \epsilon} + (t-k) \log \frac{1-\mu}{1-(\mu-\epsilon)} , \tag{47}$$

and $k \sim \mathrm{Binomial}(t, \mu_a)$. What we want to show is that as μ_a lowers, the probability that $\Lambda_t \geq \log(1/\alpha)$ lowers as well, which is to be expected since k will likely be lower. Assume then we declare $\mathcal{H}_1$, so departing from (47) we can write

$$\Lambda_t = k\lambda_0 + (k-t)\lambda_1 \geq \log(1/\alpha) ,$$

where $\lambda_0 = \log \frac{\mu}{\mu-\epsilon}$ and $\lambda_1 = \log \frac{1-\mu+\epsilon}{1-\mu}$ are both positive. Since $\log(1/\alpha)$ is also positive we have $k/t \geq \frac{\lambda_1}{\lambda_0+\lambda_1} = \left(1 + \frac{\lambda_0}{\lambda_1}\right)^{-1}$. From the definition of λ_0 and λ_1 it yields

$$\begin{aligned} \frac{k}{t} &\geq \left(1 + \frac{\lambda_0}{\lambda_1}\right)^{-1} = \left(1 + \frac{\log \frac{\mu}{\mu-\epsilon}}{\log \frac{1-\mu+\epsilon}{1-\mu}}\right)^{-1} \\ &\geq \left(1 + \frac{\frac{\mu}{\mu-\epsilon} - 1}{1 - \frac{1-\mu}{1-\mu+\epsilon}}\right)^{-1} = \left(1 + \frac{\frac{\epsilon}{\mu-\epsilon}}{\frac{\epsilon}{1-\mu+\epsilon}}\right)^{-1} \qquad (48) \\ &= \left(\frac{1}{\mu-\epsilon}\right)^{-1} = \mu - \epsilon > \mu_a , \qquad (49) \end{aligned}$$

where the inequality in (48) follows from the usual bounds of the logarithm $1 - 1/x \leq \log(x) \leq x - 1$. Rearranging (49) we get $k - t\mu_a > 0$. Then, the derivative of $f_{\mu_a}(a_1, \ldots, a_t)$ takes the form

$$\begin{aligned} &\frac{d}{d\mu_a} f_{\mu_a}(a_1, \ldots, a_t) = \frac{d}{d\mu_a} \binom{t}{k} \mu_a^k (1-\mu_a)^{t-k} \\ &= \binom{t}{k} \left(k\mu_a^{k-1}(1-\mu_a)^{t-k} + \mu_a^k(-1)(t-k)(1-\mu_a)^{t-k-1}\right) \\ &= \binom{t}{k} \mu_a^{k-1}(1-\mu_a)^{t-k-1}(k - t\mu_a) > 0 , \qquad (50) \end{aligned}$$

where the last inequality stems from (49). Putting (40), (46), and (50) together results in $P_{\mu_a}(Q_S) \geq 1 - \alpha$ for all $\mu_a \leq \mu - \epsilon$. □

Lemma 3. *For a fixed arm a of parameter μ_a, consider the sequential probability ratio test defined by* (7)–(9)*, where μ, ϵ and α are given. Then, if the alternative $\mathcal{H}_1$ is true, the test is expected to terminate after T steps, with*

$$\mathbb{E}[T] \leq 1 + \frac{\log(1/\alpha)}{\mathrm{kl}(\mu, \mu-\epsilon)} , \tag{51}$$

where $\mathrm{kl}(\mu, \mu-\epsilon)$ is the Kullback-Leibler divergence between Bernoulli distributions

$$\mathrm{kl}(\mu, \mu-\epsilon) = \mu \log \frac{\mu}{\mu-\epsilon} + (1-\mu) \log \frac{1-\mu}{1-\mu+\epsilon} .$$

Proof. Let T be the smallest integer for which the test leads to the acceptance of H_1. Such variable is well defined and finite as a result of Lemma 2. Consider the sequence of damage up to time t $(d_t)_{t=1}^T$, and let $\mathbb{E}_{\mu_a}[\cdot]$ be the expectation with respect to the true distribution of the damage data (that is, $D \sim \mathrm{Bernoulli}(\mu_a)$). Then

$$\begin{aligned} \mathbb{E}_{\mu_a}[\Lambda_T] &= \mathbb{E}_{\mu_a}\left[\sum_{t=1}^T \log \frac{f_\mu(d_t)}{f_{\mu-\epsilon}(d_t)}\right] \\ &= \mathbb{E}[T]\mathbb{E}_{\mu_a}\left[\log \frac{f_\mu(d)}{f_{\mu-\epsilon}(d)}\right] = \mathbb{E}[T] R_a , \qquad (52) \end{aligned}$$

with $R_a = \mathbb{E}_{\mu_a}\left[\log \frac{f_\mu(d)}{f_{\mu-\epsilon}(d)}\right]$. Here we used Wald's identity [42] in the second equality. Furthermore

$$\begin{aligned} \mathbb{E}_{\mu_a}[\Lambda_T] &= \mathbb{E}_{\mu_a}\left[\Lambda_{T-1} + \log \frac{f_\mu(d_T)}{f_{\mu-\epsilon}(d_T)}\right] \\ &= \mathbb{E}_{\mu_a}[\Lambda_{T-1}] + R_a \leq \log(1/\alpha) + R_a . \qquad (53) \end{aligned}$$

Combining (52) and (53):

$$\mathbb{E}[T] \leq 1 + \frac{\log(1/\alpha)}{R_a} \leq 1 + \frac{\log(1/\alpha)}{\mathrm{kl}(\mu, \mu-\epsilon)} ,$$

in virtue of $R_a \geq \mathrm{kl}(\mu, \mu-\epsilon)$ for $\mu_a > \mu - \epsilon$, as was shown in (44). □

C. Proof of Theorem 3

By lemma 3, we have

$$\mathbb{E}[E_T] = \sum_{a=1}^M \mathbb{E}[T_a] \leq M\left(1 + \frac{\log(1/\alpha)}{\mathrm{kl}(\mu, \mu-\epsilon)}\right) .$$

Then by Wald's identity [42], we have

$$\mathbb{E}[E_T] = \mathbb{E}\left[\sum_{t=1}^T \mathbb{E}[\mathbb{1}\{\mu_{A_t} > \mu\}]\right] \geq \mathbb{E}\left[T \frac{1}{K-M+1}\right] ,$$

where the second inequality is due to the fact that before T, the probability of sampling an unsafe machine is at least $1/(K-M+1)$. Recalling the upper bound for $\mathbb{E}[E_T]$ in Theorem 3, one obtain

$$\mathbb{E}[T] \leq M(K-M+1)\left(1 + \frac{\log(1/\alpha)}{\mathrm{kl}(\mu, \mu-\epsilon)}\right) .$$

D. Proof of Theorem 7

We prove Theorem 7 in three steps:

1) Reformulation of Algorithm 5: We reformulate Algorithm 5 as in Algorithm 7.

In the reformulated Algorithm 7, the sampling process is independent of B-function: At each iteration τ, an (s_τ, a_τ) pair is drawn uniformly from $\mathcal{S} \times \mathcal{A}$ and then a transition $(s_\tau, a_\tau, s'_\tau, d_\tau)$ is sampled according to the MDP, and the algorithm decides whether to accept such a sample depending on the value of $B(s_\tau, a_\tau)$. When we restrict ourselves to the trajectory of samples that are accepted, i.e.

$$\{(s_\tau, a_\tau, s'_\tau, d_\tau) : B^{(\tau)}(s_\tau, a_\tau) \neq -\infty, \ \tau = 0, 1, \cdots\} ,$$

this trajectory is also a sampled trajectory of original Algorithm 5. More importantly, for such a trajectory, the probability it appears in original Algorithm 5 is the same as the probability it appears as the accepted trajectory in Algorithm 7. With that, we define

$$T_r := \min\{\tau : B^{(\tau)} = B^*\} , \tag{54}$$

Algorithm 7: Barrier Learner Algorithm Reformulated

```
Data: Constrained Markov Decision Process $\mathcal{M}$
Initialize $B^{(0)}(s,a) = 0, \forall(s,a) \in \mathcal{S}\times\mathcal{A}$
for $\tau = 0, 1, \cdots$ do
    Draw $(s_\tau, a_\tau) \sim \mathrm{Unif}(\mathcal{S}\times\mathcal{A})$
    Sample transition $(s_\tau, a_\tau, s'_\tau, d_\tau)$ according to
      $P(S_1 = s'_\tau, D_1 = d_\tau | S_0 = s_\tau, A_0 = a_\tau)$
    if $B^{(\tau)}(s_\tau, a_\tau) \neq -\infty$ then
        $B^{(\tau+1)} \leftarrow$
          barrier_update($B^{(\tau)}, s_\tau, a_\tau, s'_\tau, d_\tau$)
    else
        $B^{(\tau+1)} \leftarrow B^{(\tau)}$
    end
end
```

i.e. the earliest time when Algorithm 7 detects all unsafe state-action pairs, then we have

$$\mathbb{E}[T] = \mathbb{E}\left[\sum_{\tau=1}^{T_r} \mathbb{1}\{B^{(\tau)}(s_\tau, a_\tau) \neq -\infty\}\right], \tag{55}$$

where T is the earliest time when Algorithm 5 detects all unsafe state-action pairs, as defined in Theorem 7. Expectations are taken with respect to the respective sampling processes of Algorithm 5 and 7, which are different. With (55), it suffices to analyze the expected detection time of Algorithm 7.

2) Construction of modified algorithm: As discussed in Section III-C, an (s,a) pair is unsafe if either it causes damage immediately or it transitions to an unsafe state with non-zero probability. If only the latter happens for such an unsafe (s,a), then to be able to declare it unsafe, one must have already declared one of its succeeding states unsafe. To make such intuition precise, we recursively define disjoint subsets $\mathcal{S}_l, l = 1, 2, \cdots$ of the state space $\mathcal{S}$ as follow,

$$\mathcal{S}_1 := \{s \in \mathcal{S} : P_\pi(D_1 = 1 | S_0 = s) > 0, \forall\pi\},$$
$$\mathcal{S}_l := \begin{cases} \mathcal{S}'_l, & \mathcal{S}'_l \neq \emptyset \\ \mathcal{S} \setminus \bigcup_{k<l} \mathcal{S}_k, & \mathcal{S}'_l = \emptyset \end{cases}, \tag{56}$$

where

$$\mathcal{S}'_l = \left\{ s \in \mathcal{S} \setminus \bigcup_{k<l} \mathcal{S}_k : P_\pi\left(S_1 \in \bigcup_{k<l} \mathcal{S}_k | S_0 = s\right) > 0, \forall\pi \right\} \tag{57}$$

It is clear that given any finite MDP, $L := \max\{l > 0 : \mathcal{S}_{l+1} \neq \emptyset\}$ is finite and we define L to be the *lag* of the MDP. Following the definition (56), $\{\mathcal{S}_l, l = 1, \cdots, L, L+1\}$ is a partition of $\mathcal{S}$. Any state $s_0 \in \bigcup_{l=1}^{L} \mathcal{S}_l$ is unsafe because under any policy π, with non-zero probability it happens that starting from s_0, the MDP eventually reaches a state in $\mathcal{S}_1$ then causes damage. Furthermore, any state $s_0 \in \mathcal{S}_{L+1} := \mathcal{S} \setminus \bigcup_{l=1}^{L} \mathcal{S}_l$ is safe since $\mathcal{S}'_{L+1} = \emptyset$ implies that there exists $a_0 \in \mathcal{A}$ such that $P(S_1 \in \bigcup_{k<L+1} \mathcal{S}_k | S_0 = s_0, A_0 = a_0) = 0$, i.e. taking action a_0 keeps the MDP away from the unsafe states in $\bigcup_{l=1}^{L} \mathcal{S}_l$. However, we note that a safe state can have unsafe actions and they can be detected by the Barrier Learner Algorithm.

More importantly, the unsafe state sets $\mathcal{S}_l, l = 1, 2, \cdots, L$ satisfies that if all states in $\bigcup_{k<l} \mathcal{S}_k$ has been declared unsafe, any state-action pair in $\{(s,a) : s \in \mathcal{S}_l, a \in \mathcal{A}\}$, when sampled, can be declared unsafe with non-zero probability. Base on this property, we construct an modified barrier learning algorithm using the prior information on $\mathcal{S}_l, l = 1, 2, \cdots, L$. The modified algorithm is described in Algorithm 8.

Algorithm 8: Modified Barrier Learner Algorithm with Prior Information on $\mathcal{S}_l, l = 1, 2, \cdots, L, L+1$

```
Data: Constrained Markov Decision Process $\mathcal{M}$,
      $\mathcal{S}_l, l = 1, 2, \cdots, L, L+1$ defined for $\mathcal{M}$
Initialize $\hat{B}^{(0)}(s,a) = 0, \forall(s,a) \in \mathcal{S}\times\mathcal{A}$
Initialize $l = 1$
for $\tau = 0, 1, \cdots$ do
    Draw $(s_\tau, a_\tau) \sim \mathrm{Unif}(\mathcal{S}\times\mathcal{A})$
    Sample transition $(s_\tau, a_\tau, s'_\tau, d_\tau)$ according to
      $P(S_1 = s'_\tau, D_1 = d_\tau | S_0 = s_\tau, A_0 = a_\tau)$
    if $\hat{B}^{(\tau)}(s_\tau, a_\tau) \neq -\infty$ and $s_\tau \in \mathcal{S}_l$ then
        $\hat{B}^{(\tau+1)} \leftarrow$
          barrier_update($\hat{B}^{(\tau)}, s_\tau, a_\tau, s'_\tau, d_\tau$)
    else
        $\hat{B}^{(\tau+1)} \leftarrow \hat{B}^{(\tau)}$
    end
    if $\hat{B}^{\tau+1}(s,a) = -\infty, \forall s \in \mathcal{S}_l, a \in \mathcal{A}$ then
        $l \leftarrow l + 1$
    end
end
```

The modified algorithm is similar to Algorithm 7 but it learns $\mathcal{S}_l, l = 1, 2, \cdots, L$ in order: At the beginning ($l = 1$), it only declares (s,a) pairs associated with $\mathcal{S}_1$ unsafe until all states in $\mathcal{S}_1$ are declared unsafe, after which l increases to 2. Now the algorithm only declares (s,a) pairs associated with $\mathcal{S}_2$ unsafe. Finally after all states in $\bigcup_{l=1}^{L} \mathcal{S}_l$ are declared unsafe ($l = L+1$), the algorithm starts to learn the unsafe transitions for safe states in $\mathcal{S}_{L+1}$. Similarly, we define

$$\hat{T}_r := \min\{\tau : \hat{B}^{(\tau)} = B^*\}, \tag{58}$$

i.e. the earliest time when Algorithm 8 detects all unsafe state-action pairs.

Since the modified algorithm is more restrictive on declaring unsafe state-action pair, the expected detection time of the modified algorithm is no less than that of Algorithm 7, as stated in the following claim.

Claim 1. *Given an MDP, let T_r and $\hat{T}_r$ be the earliest times when Algorithm 7 and Algorithm 8, detect all unsafe state-action pairs in this MDP, respectively, as defined in* (54) *and* (58). *Then*

$$\mathbb{E}\left[\sum_{\tau=1}^{T_r} \mathbb{1}\{B^{(\tau)}(s_\tau, a_\tau) \neq -\infty\}\right] \leq \mathbb{E}\left[\sum_{\tau=1}^{\hat{T}_r} \mathbb{1}\{\hat{B}^{(\tau)}(s_\tau, a_\tau) \neq -\infty\}\right],$$

where the expectation is taken with respect to the sampling of $\{(s_\tau, a_\tau, s'_\tau, d_\tau), \tau = 0, 1, \cdots\}$.

Proof. Condition on a fixed sample trajectory $\mathcal{T} := \{(s_\tau, a_\tau, s'_\tau, d_\tau)\}_{\tau=0}^{\infty}$, the functions $B^{(\tau)}$ and $\hat{B}^{(\tau)}$ are deterministic. We have

$$B^{(\tau)}(s,a)|\mathcal{T} \leq \hat{B}^{(\tau)}(s,a)|\mathcal{T}, \forall \tau \geq 0, \forall (s,a) \in \mathcal{S}\times\mathcal{A}\,, \quad (59)$$

proved by induction: we have

$$B^{(0)}(s,a)|\mathcal{T} \leq \hat{B}^{(0)}(s,a)|\mathcal{T}, \forall (s,a) \in \mathcal{S}\times\mathcal{A}\,,$$

at initialization. Suppose that (59) holds at time $\tau = t$. If (s_t, a_t, s'_t, d_t) is accepted by both algorithms, or rejected by both algorithms, we have

$$B^{(t+1)}(s,a)|\mathcal{T} \leq \hat{B}^{(t+1)}(s,a)|\mathcal{T}, \forall (s,a) \in \mathcal{S}\times\mathcal{A}\,. \quad (60)$$

If (s_t, a_t, s'_t, d_t) is rejected by Algorithm 7 and accepted by Algorithm 8, then we have

$$B^{(t)}(s_t,a_t) = -\infty, \hat{B}^{(t)}(s_t,a_t) = 0\,.$$

(60) still holds, since only $\hat{B}^{(t+1)}(s_t,a_t)$ is updated to either 0 or $-\infty$. If (s_t, a_t, s'_t, d_t) is accepted by Algorithm 7 and rejected by Algorithm 8, then we have

$$B^{(t)}(s_t,a_t) = \hat{B}^{(t)}(s_t,a_t) = 0\,.$$

(60) still holds, since only $B^{(t+1)}(s_t,a_t)$ is updated to either 0 or $-\infty$. Now from (59), we immediately know that condition on the fixed sample trajectory $\mathcal{T}$,

$$\mathbb{1}\{B(s_\tau,a_\tau) \neq -\infty\} \leq \mathbb{1}\{\hat{B}(s_\tau,a_\tau) \neq -\infty\}, \forall \tau = 0, 1, \cdots$$

Notice that T_r ($\hat{T}_r$) is the minimum t such that $B^{(t)}$ ($\hat{B}^{(t)}$) becomes exactly the same as B^*. Then $T_r|\mathcal{T} \leq \hat{T}_r|\mathcal{T}$. Therefore one have, by law of total expectation,

$$\begin{aligned}
&\mathbb{E}\left[\sum_{\tau=0}^{T_r} \mathbb{1}\{B(s_\tau,a_\tau) \neq -\infty\}\right] \\
=\,& \mathbb{E}\left[\mathbb{E}\left[\sum_{\tau=0}^{T_r} \mathbb{1}\{B(s_\tau,a_\tau) \neq -\infty\}\middle|\mathcal{T}\right]\right] \\
\leq\,& \mathbb{E}\left[\mathbb{E}\left[\sum_{\tau=0}^{\hat{T}_r} \mathbb{1}\{\hat{B}(s_\tau,a_\tau) \neq -\infty\}\middle|\mathcal{T}\right]\right] \\
=\,& \mathbb{E}\left[\sum_{\tau=0}^{\hat{T}_r} \mathbb{1}\{\hat{B}(s_\tau,a_\tau) \neq -\infty\}\right]
\end{aligned}$$

□

3) Expected detection time of modified algorithm: Lastly, we prove the following Theorem regarding the expected detection time of the modified algorithm.

Theorem 8. *Given an MDP with $\mathcal{S}_l, l = 1, 2, \cdots, L, L+1$ defined as in* (56)*. Assume that exists $\mu > 0$ such that the transition probability $P(S_1 = s'|S_0 = s, A_0 = a)$, is either zero or lower bounded by μ, for all $s, s' \in \mathcal{S}, a \in \mathcal{A}$. Let $\hat{T}_r$ be earliest time when Algorithm 8 detects all unsafe state-action pairs as defined in* (58)*, then we have*

$$\mathbb{E}\left[\sum_{\tau=1}^{\hat{T}_r} \mathbb{1}\{\hat{B}^{(\tau)}(s_\tau,a_\tau) \neq -\infty\}\right] \leq \frac{|\mathcal{S}||\mathcal{A}|}{\mu}\sum_{l=1}^{L+1}\left(\sum_{k=1}^{|\mathcal{S}_l||\mathcal{A}|}\frac{1}{k}\right).$$

Proof. Let $\hat{T}_l, l = 1, \cdots, L+1$ denote the earliest time when all unsafe state-action pairs associated with $\mathcal{S}_l$ are detected by Algorithm 8, and we let $\hat{T}_0 = 0$. Then clearly

$$\hat{T}_{l-1} < \hat{T}_l, l = 1, \cdots, L+1\,, \quad \hat{T}_{L+1} = \hat{T}_r\,,$$

and $\Delta_l := \sum_{t=\hat{T}_{l-1}}^{\hat{T}_l - 1} \mathbb{1}\{\hat{B}^{(t)}(s_t,a_t) \neq -\infty\}$ is the number of accepted samples by Algorithm 8 between $\hat{T}_{l-1}$ and $\hat{T}_l$.

Notice that we can view the barrier learning process between $\hat{T}_{l-1}$ and $\hat{T}_l$ as detecting unsafe machines in the safe multi-arm bandits problem discussed in Section II: At time $\hat{T}_{l-1}$, there are in total $K_l := \left|\left\{(s,a) : s \in \bigcup_{k=l}^{L+1}\mathcal{S}_k, a \in \mathcal{A}\right\}\right|$ machines, and the number of unsafe machines is

$$M_l := |\{(s,a) : s \in \mathcal{S}_l, a \in \mathcal{A}, B^*(s,a) = -\infty\}|\,.$$

We have $K_l \leq |\mathcal{S}||\mathcal{A}|$, $M_l \leq |\mathcal{S}_l||\mathcal{A}|$. Furthermore, condition on such an unsafe machine is pulled, i.e. an unsafe (s,a) in $\mathcal{S}_l$ is accepted by Algorithm 8, the probability of declaring it unsafe is at least μ. Because the (s,a) pair either transitions to some $s \in \bigcup_{k=1}^{l-1}\mathcal{S}_k$ that has been declared unsafe or directly incurs damage with non-zero probability, and that probability is lower bounded by μ according to our assumption.

The acceptance of sample $(s_\tau, a_\tau, s'_\tau, d_\tau)$ is equivalent to pulling a uniformly randomly drawn arm out of arms that have not been declared unsafe. Theorem 2 suggests that the expected number of such "pulling" is upper bounded as

$$\mathbb{E}[\Delta_l] \leq \frac{K_l}{\mu}\left(\sum_{k=1}^{M_l}\frac{1}{k}\right) \leq \frac{|\mathcal{S}||\mathcal{A}|}{\mu}\left(\sum_{k=1}^{|\mathcal{S}_l||\mathcal{A}|}\frac{1}{k}\right).$$

Finally, we have

$$\begin{aligned}
&\mathbb{E}\left[\sum_{t=0}^{\hat{T}_r} \mathbb{1}\{\hat{B}^{(t)}(s_t,a_t) \neq -\infty\}\right] \\
=\,& \mathbb{E}\left[\sum_{l=1}^{L+1}\Delta_l\right] \leq \frac{|\mathcal{S}||\mathcal{A}|}{\mu}\sum_{l=1}^{L+1}\left(\sum_{k=1}^{|\mathcal{S}_l||\mathcal{A}|}\frac{1}{k}\right).
\end{aligned}$$

□

Now we are ready to prove Theorem 7.

Proof of Theorem 7. Given any MDP, we have $|\mathcal{S}_l| \leq |\mathcal{S}|, \forall l = 0, 1, \cdots, L+1$, then we have

$$\begin{aligned}
\mathbb{E}[T] \leq\,& \mathbb{E}\left[\sum_{\tau=1}^{\hat{T}_r} \mathbb{1}\{\hat{B}^{(\tau)}(s_\tau,a_\tau) \neq -\infty\}\right] \\
&\leq \frac{|\mathcal{S}||\mathcal{A}|}{\mu}\sum_{l=1}^{L+1}\left(\sum_{k=1}^{|\mathcal{S}_l||\mathcal{A}|}\frac{1}{k}\right) \leq \frac{|\mathcal{S}||\mathcal{A}|(L+1)}{\mu}\left(\sum_{k=1}^{|\mathcal{S}||\mathcal{A}|}\frac{1}{k}\right),
\end{aligned}$$

where the first equality is from (55) and Claim 1. □

References

[1] A. Castellano, J. Bazerque, and E. Mallada, "Learning to be safe, in finite time," in *2021 American Control Conference (ACC)*, IEEE, 2021.

[2] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural computation*, vol. 29, no. 9, pp. 2352–2449, 2017.

[3] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[4] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, p. 484, 2016.

[5] W. M. Haddad and V. Chellaboina, *Nonlinear Dynamical Systems and Control: A Lyapunov-Based Approach*. Princeton University Press, 2011.

[6] K. Zhou and J. C. Doyle, *Essentials of Robust Control*, vol. 104. Prentice hall Upper Saddle River, NJ, 1998.

[7] F. Berkenkamp, M. Turchetta, A. P. Schoellig, and A. Krause, "Safe Model-based Reinforcement Learning with Stability Guarantees," *arXiv*, 2017.

[8] T. J. Perkins and A. G. Barto, "Lyapunov design for safe reinforcement learning," *Journal of Machine Learning Research*, vol. 3, no. Dec, pp. 803—832, 2002.

[9] M. Jin and J. Lavaei, "Control-Theoretic Analysis of Smoothness for Stability-Certified Reinforcement Learning," *2018 IEEE Conference on Decision and Control (CDC)*, vol. 00, pp. 6840–6847, 2018.

[10] Y. Chow, O. Nachum, E. Duenez-Guzman, and M. Ghavamzadeh, "A Lyapunov-based Approach to Safe Reinforcement Learning," *arXiv*, 2018.

[11] S. Singh, S. M. Richards, V. Sindhwani, J.-J. E. Slotine, and M. Pavone, "Learning Stabilizable Nonlinear Dynamics with Contraction-Based Regularization," *arXiv*, 2019.

[12] J. Morimoto and K. Doya, "Robust Reinforcement Learning," *Neural Computation*, vol. 17, no. 2, pp. 335—359, 2005.

[13] C. W. Anderson, P. M. Young, M. R. Buehner, J. N. Knight, K. A. Bush, and D. C. Hittle, "Robust reinforcement learning control using integral quadratic constraints for recurrent neural networks," *IEEE Transactions on Neural Networks*, vol. 18, no. 4, pp. 993—1002, 2007.

[14] F. Berkenkamp and A. P. Schoellig, "Safe and robust learning control with Gaussian processes," in *2015 European Control Conference (ECC)*, pp. 2496—2501, 2015.

[15] S. R. Friedrich and M. Buss, "A robust stability approach to robot reinforcement learning based on a parameterization of stabilizing controllers," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3365—3372, 2017.

[16] L. Buşoniu, T. d. Bruin, D. Tolić, J. Kober, and I. Palunko, "Reinforcement learning for control: Performance, stability, and deep approximators," *Annual Reviews in Control*, vol. 46, pp. 8–28, 2018.

[17] P. L. Donti, M. Roderick, M. Fazlyab, and J. Z. Kolter, "Enforcing robust control guarantees within neural network policies," in *International Conference on Learning Representations*, arXiv, 2020.

[18] J. Garcia and F. Fernandez, "A Comprehensive Survey on Safe Reinforcement Learning," *Journal of Machine Learning Research*, vol. 16, no. 1, pp. 1437—1480, 2015.

[19] P. Geibel, "Reinforcement Learning for MDPs with Constraints," vol. 4212 of *Machine Learning: ECML 2006*, pp. 646–653, 2006.

[20] S. Junges, N. Jansen, C. Dehnert, U. Topcu, and J.-P. Katoen, "Safety-Constrained Reinforcement Learning for MDPs," *arXiv*, 2015.

[21] M. Turchetta, F. Berkenkamp, and A. Krause, "Safe Exploration in Finite Markov Decision Processes with Gaussian Processes," *arXiv*, 2016.

[22] G. Dalal, K. Dvijotham, M. Vecerik, T. Hester, C. Paduraru, and Y. Tassa, "Safe Exploration in Continuous Action Spaces," *arXiv*, 2018.

[23] A. Wachi, Y. Sui, Y. Yue, and M. Ono, "Safe Exploration and Optimization of Constrained MDPs Using Gaussian Processes," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018.

[24] T. Xu, Y. Liang, and G. Lan, "A Primal Approach to Constrained Policy Optimization: Global Optimality and Finite-Time Analysis," *arXiv*, 2020.

[25] S. Paternain, M. Calvo-Fullana, L. F. O. Chamon, and A. Ribeiro, "Safe Policies for Reinforcement Learning via Primal-Dual Methods," *arxiv*, 11 2019.

[26] D. Ding, X. Wei, Z. Yang, Z. Wang, and M. R. Jovanović, "Provably Efficient Safe Exploration via Primal-Dual Policy Optimization," *arXiv*, 2020.

[27] Y. Ge, F. Zhu, X. Ling, and Q. Liu, "Safe Q-Learning Method Based on Constrained Markov Decision Processes," *IEEE Access*, vol. 7, pp. 165007–165017, 2019.

[28] A. Wachi and Y. Sui, "Safe Reinforcement Learning in Constrained Markov Decision Processes," *arXiv*, 2020.

[29] A. Hasanzade Zonuzy, A. Bura, D. Kalathil, and S. Shakkottai, "Learning with Safety Constraints: Sample Complexity of Reinforcement Learning for Constrained MDPs," *arXiv*, 2020.

[30] T. Lattimore and C. Szepesvári, *Bandit algorithms*. Cambridge University Press, 2020.

[31] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[32] C. J. C. H. Watkins, "Learning from delayed rewards," 1989.

[33] M. G. Azar, R. Munos, M. Ghavamzadeh, and H. J. Kappen, "Speedy q-learning," 2011.

[34] A. M. Devraj and S. P. Meyn, "Zap q-learning," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 2232–2241, 2017.

[35] A. Wald, "Sequential Tests of Statistical Hypotheses," *The Annals of Mathematical Statistics*, vol. 2, no. 16, pp. 117—186, 1945.

[36] T. Lai and H. Robbins, "Asymptotically efficient adaptive allocation rules," *Advances in Applied Mathematics*, vol. 6, no. 1, pp. 4–22, 1985.

[37] D. Ding, K. Zhang, T. Basar, and M. Jovanovic, "Natural policy gradient primal-dual method for constrained markov decision processes," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[38] G. B. Folland, *Real analysis: modern techniques and their applications*, vol. 40. John Wiley & Sons, 1999.

[39] D. P. Bertsekas *et al.*, *Dynamic programming and optimal control: Vol. II*. Athena scientific, Belmont, 2012.

[40] H. Robbins and S. Monro, "A Stochastic Approximation Method," *The Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400 – 407, 1951.

[41] J. N. Tsitsiklis, "Asynchronous stochastic approximation and q-learning," *Machine learning*, vol. 16, no. 3, pp. 185–202, 1994.

[42] A. Wald, "On Cumulative Sums of Random Variables," *The Annals of Mathematical Statistics*, vol. 15, no. 3, pp. 283 – 296, 1944.