Safety-Guaranteed, Accelerated Learning in MDPs with Local Side Information*

Pranay Thangeda and Melkior Ornik

Abstract— **In environments with uncertain dynamics, synthesis of optimal control policies mandates exploration. The applicability of classical learning algorithms to real-world problems is often limited by the number of time steps required for learning the environment model. Given some local side information about the differences in transition probabilities of the states, potentially obtained from the agent's onboard sensors, we generalize the idea of indirect sampling for accelerated learning to propose an algorithm that balances between exploration and exploitation. We formalize this idea by introducing the notion of the value of information in the context of a Markov decision process with unknown transition probabilities, as a measure of the expected improvement in the agent's current estimate of transition probabilities by taking a particular action. By exploiting available local side information and maximizing the estimated value of learned information at each time step, we accelerate the learning process and subsequent synthesis of the optimal control policy. Further, we define the notion of agent safety, a vital consideration for physical systems, in the context of our problem. Under certain assumptions, we provide guarantees on the safety of an agent exploring with our algorithm that exploits local side information. We illustrate agent safety and the improvement in learning speed using numerical experiments in the setting of a Mars rover, with data from onboard sensors acting as the local side information.**

I. Introduction

The problem of controlling a physical system in an unknown environment is of significant practical interest [1], [2]. The agent is required to learn, in some way, the underlying dynamics of the environment before planning an optimal solution for its objective. Learning in unknown environments has been studied extensively and several approaches exist in the literature [3], [4], [5]. Environments with unknown dynamics are often modeled as Markov decision processes (MDPs). In the MDP framework, existing algorithms such as those in the class of *Probably approximately correct in Markov decision processes (PAC-MDP)* [6], [7] and *Bayesian exploration bonus (BEB)* [8] find the optimal policy by setting up a trade-off between exploring previously unvisited states to gain additional information about the environment and exploiting the current knowledge to maximize reward. However, these algorithms do not use any prior or side information about the transition probabilities thereby requiring an impractical number of samples to build an accurate model of real-world problems.

*This work was supported by NASA's Space Technology Research Grants program for Early Stage Innovations under the grant 'Safety-Constrained and Efficient Learning for Resilient Autonomous Space Systems.'

Pranay Thangeda and Melkior Ornik are with the Department of Aerospace Engineering and the Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, Urbana, USA. {pranayt2, mornik}@illinois.edu.

In order to expedite learning, [3] proposes an algorithm that uses additional information in the form of bounds on differences between transition probabilities of states to partly reuse the collected samples. In other words, every sample collected by an agent is counted directly at its current state and, with some weight, at other states that have similar transition probabilities. While such an approach improves the sample collection rate, the assumption of prior knowledge about similarity between *all* the states is unrealistic in most problems. For an agent operating in an environment that requires it to perform tasks autonomously, it is rational to assume that the side information can only be gathered using limited onboard resources. We illustrate this by considering the environment of a Mars rover, a setting we use as a running example throughout this paper. In this case, the information assumed by [3] would need to be available a priori for the entire environment, whereas the local side information is based on the data the rover collects online using its onboard sensors. Side information, if available, is usually restricted to the immediate neighborhood of the agent.

Exploration algorithms, while learning the state transition probabilities by visiting all state-action pairs, fail to account for the safety of the agent. Safety is an important consideration, particularly for agents operating in physical environments where even a single unsafe action could be harmful not only to the agent but also to the other elements in the environment. While safety during learning is an active area of research and several different notions of safety have been studied [9], [10], [11], guarantees of safety are either provided by explicitly considering safety in the optimality criterion or by assuming some external knowledge or risk metric, both of which limit their generalizability.

The two central contributions of this paper are as follows. First, we generalize the framework of indirect sampling proposed in [3] to environments where agents have access to local side information and propose an exploration bonus that maximizes the value of information gain at each state during learning. The second important contribution is a definition of safety relevant to the context of an agent with local side information with guarantees on its safety while operating with the proposed algorithm.

Notation: For a finite set $\mathcal{S}$, $|\mathcal{S}|$ denotes its cardinality, for a subset $\mathcal{D} \subset S$, $\mathcal{D}^c$ denotes its complement; $\mathcal{D}^c = \{s \in \mathcal{S} | s \notin \mathcal{D}\}$. For $x, y \in \mathbb{R}^n$, $\|x\|_p$ denotes the p-norm, and $d(x,y)$ denotes the Euclidean distance between x and y.

II. Preliminaries

We consider problems where the real environment is specified by a finite horizon, undiscounted Markov Decision Process (MDP) $\mathcal{M} = (H, \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$. $H \geq 0$ is the time horizon, $\mathcal{S}$ is the finite state space, $\mathcal{A}$ is the finite action space, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0,1]$ is the unknown state transition probability function, and $\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow [0, \bar{R}]$ is the reward function which we, without loss of generality, assume to be known. At any state $s \in \mathcal{S}$, the agent takes an action $a \in \mathcal{A}$ and proceeds to next state based on the distribution $\mathcal{P}(.|s,a)$ while collecting a reward $\mathcal{R}(s,a)$. A policy π is a mapping from states to actions that specifies a decision-making strategy. The *state-value* of a policy for a state is defined as the expected sum of rewards over the next H time steps $V_H^\pi(s) = E\left[\sum_{t=1}^{H} \mathcal{R}(s_t, \pi(s_t)) | s_1 = s\right]$ and the *action value* is defined as the expected sum of rewards by taking an action a and then following the policy $Q_H^\pi(s,a) = \mathcal{R}(s,a) + E\left[\sum_{t=2}^{H} \mathcal{R}(s_t, \pi(s_t))\right]$. Further, the state value can be expressed in the recursive Bellman equation form

$$V_H^\pi(s) = \mathcal{R}(s, \pi(s)) + \sum_{s'} \mathcal{P}(s'|s,a) V_{H-1}^\pi(s').$$

We seek a policy that maximizes the state value. The optimal policy π^* can be obtained by solving the *Bellman's optimality equation* in state value form

$$V_H^*(s) = \max_a [\mathcal{R}(s,a) + \sum_{s'} \mathcal{P}(s'|s,a) V_{H-1}^*(s')]$$

where $\pi^*(s)$ is simply the action a that maximizes state value at s. For an MDP with known transition probabilities and reward function, the optimal state value can be obtained by solving the above equation using classical planning algorithms [12].

Algorithms such as those in the class of PAC-MDP [13] and BEB [7] achieve the balance between exploration and exploitation by complementing reward with an exploration bonus. Unknown state transition probabilities are estimated from the samples generated by the transition probability distribution during exploration. Formally, let $\alpha(s,a)$ denote the number of samples where the agent performed an action a in the state s and $\alpha(s'|s,a)$ denote the number of those instances where the resultant state was $s' \in \mathcal{S}$. Then the estimate of the transition probability $\mathcal{P}(s'|s,a)$ is given as $\hat{\mathcal{P}}(s'|s,a) = \alpha(s'|s,a)/\alpha(s,a)$. At every time step, an action taken by the agent contributes exactly one sample using the above-mentioned algorithm, referred to as *direct sampling*. A state-action pair (s,a) is considered to be *known* if $\alpha(s,a) \geq m$, for some fixed threshold $m \in \mathbb{N}$. This consideration is justified as one can use large deviation inequalities to ensure that the estimated transition probabilities closely represent the true dynamics making further sampling unnecessary [8]. Therefore, learning the environment model accurately would require collecting at least $m|\mathcal{S}||\mathcal{A}|$ samples, an impractical number in problems with large state and action spaces. The next section describes an algorithm to overcome this issue by exploiting additional information available to the agent about the states in its neighborhood.

III. Learning with Local Side Information

In this section, we introduce the notion of *local side information* and propose an algorithm that exploits it to accelerate learning in MDPs. Even in the environments where we lack the knowledge of the exact features-dynamics relationship, we could still exploit the similarity as states with similar features are likely to have similar dynamics. Given some measure of similarity between states that reflects the similarity in their dynamics, we could partially reuse an observation at multiple states thereby increasing the sample collection speed – a framework originally introduced in [3] and referred to as *indirect sampling*. To formalize the notion of side information, we first define a measure of similarity between transition probabilities of states and then use it to quantify the extent of sample reusability between states. As stated earlier, these definitions were first introduced in the author's previous work and presented here for the sake of completeness.

Definition 1. *Let $s, \bar{s} \in \mathcal{S}$, and let $0 \leq \varepsilon \leq 1$. States s and $\bar{s}$ are ε-distant if there exists a permutation $\Pi : \mathcal{S} \rightarrow \mathcal{S}$ such that $|\mathcal{P}(s'|s,a) - \mathcal{P}(\Pi(s')|\bar{s},a)| \leq \varepsilon$ for any $s' \in \mathcal{S}$ and any $a \in \mathcal{A}$.*

Intuitively, Π in the above definition encodes similar movement – in the context of our running example, if executing an action a in state s is expected to take the rover to a state s' immediately north of s, then $\Pi(s')$ is the state immediately north of $\bar{s}$. Although Π depends on both s and $\bar{s}$, to simplify notation, we emphasize the dependency on $\bar{s}$ only when necessary. We also define $\Delta : \mathcal{S} \times \mathcal{S} \rightarrow [0,1]$ so that $\Delta(s,\bar{s})$ is the smallest value such that $s, \bar{s}$ are $\Delta(s,\bar{s})$-distant. We now introduce *weighting function* in Definition 2 as a measure of sample reusability given knowledge of distance Δ between the transition probabilities of states.

Definition 2. *Function $w : [0,1] \rightarrow [0,1]$ is a* weighting function *if it is monotonically decreasing and satisfies $w(0) = 1$ and $w(1) = 0$.*

To simplify the notation, we define $\omega(s,\bar{s}) = w(\Delta(s,\bar{s}))$ for any $s, \bar{s} \in \mathcal{S}$. We assume that, given current state of the agent s, we have knowledge of $\Delta(s,\bar{s})$ for all $\bar{s} : d(s,\bar{s}) \leq \lambda$ where λ, the *observation radius*, denotes the range of agent's onboard sensors. We note that this is a generalization of the approach proposed in [3], and for a large enough observation radius the two notions are equivalent. Apart from the local side information, our framework can also utilize any prior information available to obtain an initial estimate of the weights. The prior information, for example, could be the sparse terrain characteristics data obtained from an orbiter.

The generalized algorithm for weighted indirect sampling using local side information, given weighting function ω, is presented as Algorithm 1. Any weights obtained from prior information are denoted by ϕ_0 and in the case where no prior information available, $\phi_0(s,\bar{s}) = 0 \ \forall s, \bar{s} \in \mathcal{S}$. At every time step, the algorithm uses the current state, current action, and the resultant state triad, (s,a,s'), to update the corresponding direct sample counts, $\alpha(s,a)$ and $\alpha(s'|s,a)$. The direct sample counts and the weighting function are then used to update $\#(s,a)$ and $\#(s'|s,a)$, the effective number of samples (i.e., both direct and indirect weighted) at state-action pair (s,a) and the number of such samples leading to s' or equivalent (i.e., $\Pi(s')$) state. The effective sample counts are then used to estimate the transition probabilities, as detailed in Algorithm 1. We note that irrespective of the similarity between states, the above algorithm at minimum increases the overall sample count by one, as at the current state of the agent s, $\omega(s,s)=1$. Hence, the indirect sampling algorithm collects samples at least as quickly as direct sampling. While any function satisfying Definition 2 can be used

Algorithm 1 Weighted indirect sampling

```
1: Let $\alpha(s'|s,a)=0$ for all $s,s' \in \mathcal{S}, a \in \mathcal{A}$.
2: Let $\alpha(s,a)=0$ for all $s \in \mathcal{S}, a \in \mathcal{A}$.
3: Let $\#(s'|s,a)=0$ for all $s,s' \in \mathcal{S}, a \in \mathcal{A}$.
4: Let $\#(s,a)=0$ for all $s \in \mathcal{S}, a \in \mathcal{A}$.
5: Let $\phi(s,\bar{s})=\phi_0(s,\bar{s})$ for all $s,\bar{s} \in \mathcal{S}$,
6: repeat at each time step
7:     Let $s$ be current state of the system
8:     Let $a$ be the performed action
9:     Let $s'$ be the resulting state of the system
10:    for all $\bar{s}: d(s,\bar{s}) \leq \lambda$ do
11:        $\phi(s,\bar{s})=\omega(s,\bar{s})$
12:    end for
13:    $\alpha(s'|s,a) := \alpha(s'|s,a)+1$
14:    $\alpha(s,a) := \alpha(s,a)+1$
15:    for all $\bar{s},\bar{s}' \in S$ do
16:        $\#(\Pi(s')|\bar{s},a) := \sum_{\bar{s}'} \alpha(\Pi(s')|\bar{s}',a)\phi(\bar{s},\bar{s}')$
17:        $\#(\bar{s},a) := \sum_{\bar{s}'} \alpha(\bar{s}',a)\phi(\bar{s},\bar{s}')$
18:        if $\#(\bar{s},a) > 0$ then
19:            $\hat{\mathcal{P}}(\Pi(s')|\bar{s},a) := \#(\Pi(s')|\bar{s},a)/\#(\bar{s},a)$
20:        else
21:            $\hat{\mathcal{P}}(\Pi(s')|\bar{s},a) := 0$
22:        end if
23:    end for
24: until $\alpha(s,a) \geq m \quad \forall s \in \mathcal{S}, a \in \mathcal{A}$
```

as a weighting function, selecting the right one is essential for balancing between speed-up in sample collection and error in estimated transition probabilities [3]. While weighted indirect sampling significantly reduces the error in the initial time steps when compared to direct sampling, the error converges to a non-zero value in the long term. This issue can be addressed by introducing a time-varying weighting function, $\omega_t(s,s')=w(\Delta(s,s'),t)$ where $\omega_t(s,s') \to 0$ as $t \to \infty$. ω_t, while performing better than direct sampling in short term, will also have an error that eventually converges to zero.

So far, we focused on using local side information to accelerate the collection of samples with little consideration for the actual problem of finding the optimal control policy. As discussed earlier, learning algorithms typically handle exploration-exploitation trade-off by rewarding the agent for visiting a previously unexplored or underexplored state. However, in the case of agents with local side information, not every state is the same in terms of the information gain one could achieve by visiting it. For example, visiting a state with a high degree of similarity with its neighboring states will contribute more to the overall sample count than visiting a state with the same count of current samples but little or no similarity with its neighbors. This motivates the notion of a benefit, defined as a measure of the estimated total information gain by sampling at a state. Benefit, which helps in deciding the action that adds the highest value to the learning process, should encompass:

1) Quantification of similarity of a state to other states – visiting a state with highly similar neighbors is beneficial as it adds to the sample count of its neighbors.
2) Significant bonus for visiting a previously unexplored state – visiting the state for the first time provides information on its similarity to neighboring states.
3) Higher bonus for a state-action pair with low sample count – low sample count translates to a high error in estimated transition probabilities. Note that this is the only component considered in [3].

With these requirements in mind, we propose the following benefit function:

$$\mathcal{B}(s,a)=\mu\eta(s)+\gamma\sum_{\bar{s}}\frac{\omega(s,\bar{s})}{1+\alpha(\bar{s},a)}, \tag{1}$$

where $\eta(s):\mathcal{S}\to\{0,1\}$ is a function assuming value of 1 if the state s was not previously visited and 0 if the state has already been visited, $\mu,\gamma \in \mathbb{R}_{\geq 0}$ are tunable parameters, and $\bar{s} \in \mathcal{S}$ are states such that $d(s,\bar{s}) \leq \lambda$. While η motivates the agent to actively explore previously unvisited states, the second component of (1) ensures that states with highly similar and less explored neighbors are visited first. In every element of the summation over states within observation radius, the weight $\omega(s,\bar{s})$ in the numerator signifies similarity of that state $\bar{s}$ with state under consideration s, and the sample count $\alpha(\bar{s},a)$ in the denominator prioritizes state-action pairs with less number of samples. The addition of 1 in the denominator is only to ensure that the benefit $\mathcal{B}(s,a)=\infty$ does not happen when $\alpha(\bar{s},a)=0$. Using the benefit value as the exploration bonus, the optimal action at any state would be the action that maximizes the value function

$$\hat{V}_H^*(s)=\max_a\left\{\mathcal{B}(s,a)+\mathcal{R}(s,a)+\sum_{s'\in\mathcal{S}}\hat{\mathcal{P}}(s'|s,a)\hat{V}_{H-1}^*(s')\right\},$$

where $\hat{V}_0^*(s)=\max_a\mathcal{B}(s,a)$ and $\hat{\mathcal{P}}$ denotes the current estimate of the state transition probabilities. While we omitted its timestamp to simplify notation, note that, unlike the reward function, the benefit function $\mathcal{B}(s,a)$ is time-varying. Hence, finding values of benefit at future states, required for calculating $\hat{V}_{H-1}^*(s')$, is non-trivial. We could solve this issue by either maintaining expectation of future benefits at all states or by acting as if it will be time-invariant in the future, as done in [3]. In the next section, we define safety in the context of our problem and provide safety guarantees for an agent operating in an environment with access to side information.

IV. Safety Guarantees with Side Information

In this section, we formalize the notion of safety and provide guarantees on the safety of an agent operating with the proposed learning algorithm. In a given state space $\mathcal{S}$, we define *unsafe* states as a subset of states $\mathcal{D} \subset \mathcal{S}$ that are undesirable and are a priori known to the agent. We also assume that if an agent enters an unsafe state at time t, further exploration is not possible. We consider a policy to be *safe* if an agent following the policy always takes actions that are least likely to result in it moving to an unsafe state.

Definition 3. *Policy π is* safe *if, for any safe state $s_t \in \mathcal{D}^c$, $P(s_{t+1} \in \mathcal{D} | s_t, \pi(s_t)) = \min_{a_t \in \mathcal{A}} P(s_{t+1} \in \mathcal{D} | s_t, a_t)$.*

Given a reward function that assigns an appropriately low reward for all unsafe states, an agent acting optimally with perfect knowledge of the environment will enter an unsafe state at time t with probability $\min_{a \in \mathcal{A}} \sum_{s' \in \mathcal{D}} \mathcal{P}(s'|s_t, a)$. For an agent exploring the environment, a safe policy only guarantees safety inasmuch the optimal policy guarantees safety. Therefore, for establishing the sufficient conditions for safety, a policy is safe if its outcomes at every time step are the same as the outcomes of the optimal policy. In terms of action-values, the sufficient conditions can be given as

$$\operatorname*{argmax}_a \hat{Q}^*_H(s,a) = \operatorname*{argmax}_a Q^*_H(s,a), \tag{2}$$

where $\hat{Q}^*_H(s,a)$ is the optimal action-value given current estimates of state transition probabilities and $Q^*_H(s,a)$ denotes the optimal action-value given true state transition probabilities. The outcome of a policy that maximizes $\hat{Q}^*_H(s,a)$ at a state s will be the same as the outcome of the optimal policy $\operatorname{argmax}_a Q^*_H(s,a)$ only if the error in the optimal action-values $|\max_{a \in \mathcal{A}} \hat{Q}^*_H(s,a) - \max_{a \in \mathcal{A}} Q^*_H(s,a)|$ is less than the minimum difference between the true action-values at the state $\min_{a_i, a_j \in \mathcal{A}} |Q^*_H(s,a_i) - Q^*_H(s,a_j)|$. We assume that the agent is myopic to obtain tractable error formulations.

Assumption 1. *The agent is myopic, i.e., the horizon of the agent H equals zero.*

We note that the results in Section V prove that the guarantees that we obtain under this assumption are conservative; the agent behaves safely even for a non-zero horizon. For convenience, we use $\hat{Q}^*(s,a)$ to denote $\hat{Q}^*_0(s,a)$, $Q^*(s,a)$ to denote $Q^*_0(s,a)$, and $\hat{\pi}^*$ to denote the optimal policy given current estimates. Also, we quantify the error in state transition probabilities using ε, the maximum error for any action at a state, $\varepsilon = \max_{a \in \mathcal{A}} \|\hat{\mathcal{P}}(s'|s,a) - \mathcal{P}(s'|s,a)\|_1$. Then, for a given minimum difference between true action-values at a state Q_D, we claim that an agent takes a *safe action* if the maximum error in transition probabilities at that state satisfies the relationship $\varepsilon < Q_D/\bar{R}$, where $\bar{R}$ is the maximum possible reward at any state.

Theorem 1. *If $\varepsilon < Q_D/\bar{R}$, then under Assumption 1, at any state s_t in $\mathcal{D}^c$, $P(s_{t+1} \in \mathcal{D} | s_t, \hat{\pi}^*(s_t)) = \min_{a_t \in \mathcal{A}} P(s_{t+1} \in \mathcal{D} | s_t, a_t)$.*

Proof: At any time step t, define

$$G(s_{t+1}|s_t,a_t) = \hat{\mathcal{P}}(s_{t+1}|s_t,a_t) - \mathcal{P}(s_{t+1}|s_t,a_t), \tag{3}$$

where $\hat{\mathcal{P}}$ is the current estimate of state transition probabilities. Given maximum error in transition probability for any action at that state is less than ε,

$$\sum_{s' \in \mathcal{S}} |G(s'|s_t,a)| \le \varepsilon \ \forall a \in \mathcal{A}. \tag{4}$$

The error in the action-value at any state-action pair is $\hat{Q}^*(s_t,a_t) - Q^*(s_t,a_t)$. Using (3) and the assumption that horizon $H = 0$, we could express the error in value as

$$\begin{aligned} &\hat{Q}^*(s_t,a_t) - Q^*(s_t,a_t) \\ &= \sum_{s' \in \mathcal{S}} G(s'|s_t,a_t)\mathcal{R}(s'|s_t,a_t). \end{aligned} \tag{5}$$

We know that the reward function is defined such that it satisfies $0 \le \mathcal{R} \le \bar{R}$. Using this and (4) we have

$$\begin{aligned} &\sum_{s' \in \mathcal{S}} G(s'|s_t,a_t)\mathcal{R}(s'|s_t,a_t) \\ &\le \sum_{s' \in \mathcal{S}} |G(s'|s_t,a_t)|\mathcal{R}(s'|s_t,a_t) \le \varepsilon \bar{R}. \end{aligned} \tag{6}$$

Substituting (5) in (6),

$$\hat{Q}^*(s_t,a_t) - Q^*(s_t,a_t) \le \varepsilon \bar{R}. \tag{7}$$

Further, we have

$$|\hat{V}^*(s_t) - V^*(s_t)| \le \|\hat{Q}^*(s_t,a_t) - Q^*(s_t,a_t)\|_\infty, \tag{8}$$

where $\hat{V}^*(s_t) = \max_{a \in \mathcal{A}} \hat{Q}^*(s_t,a)$ and $V^*(s_t) = \max_{a \in \mathcal{A}} Q^*(s_t,a)$. From (7) and (8) it follows that

$$|\hat{V}^*(s_t) - V^*(s_t)| \le \varepsilon \bar{R}. \tag{9}$$

We know that sufficient condition for an action to be safe is

$$|\hat{V}^*(s_t) - V^*(s_t)| < Q_D, \tag{10}$$

Therefore, from (9) and (10) it follows that an action taken by an agent at any state is safe if

$$\varepsilon < \frac{Q_D}{\bar{R}}. \qquad \square$$

Theorem 1 establishes guarantees on the safety of an agent for a given error in the transition probabilities at a state. However, direct sampling algorithms cannot provide an estimate of transition probability of a state-action pair without first directly collecting samples at that state. On the other hand, algorithms learning with indirect sampling provide estimates of transition probabilities at states within the agent's observation radius without actually visiting them. This feature motivates us to provide a guarantee on the safety of a policy generated by the proposed algorithm.

Assumption 2. *For a starting state $s_0 \in \mathcal{D}^c$, let T be the minimum number of time steps required for the agent to traverse, using any control policy, from s_0 to any $s \in \mathcal{D}$.*

In order to make the technical work simpler, we consider an environment where all the states have the same transition probabilities, up to a permutation.

Assumption 3. *Let $\Delta : \mathcal{S} \times \mathcal{S} \to [0,1]$ be the distance between transition probabilities of states, as defined in Definition 1. For any $s, \bar{s} \in \mathcal{D}^c$, $\Delta(s,\bar{s}) = 0$.*

The exploration bonus of the proposed algorithm, with proper choice of tuning parameters, ensures that all the actions are sampled roughly the same number of times. Hence, for an agent learning with the proposed algorithm, the number of samples collected after time T for any action is no less than $\lfloor T/|\mathcal{A}| \rfloor$. Under these assumptions, we provide an exponential bound on the safety of an agent.

Theorem 2. *Under Assumptions 1, 2, and 3, at any time step t such that $s_t \in \mathcal{D}^c$, $P(s_{t+1} \in \mathcal{D} | s_t, \hat{\pi}^*(s_t)) \leq e^{-\frac{2\lfloor T/|\mathcal{A}|\rfloor Q_D^2}{|S|\bar{R}^2}}$.*

Proof: For any $t < T$, the proof is trivial as the probability of taking an unsafe action is zero. For the case $t \geq T$, the proof follows from Theorem 1 and Hoeffding's inequality [14]. Under Assumption 2, for any starting state s_0, the agent collects at least T samples before reaching the neighborhood of an unsafe state. Under Assumption 3, the transition probability of an action remains the same at any state $s \in \mathcal{D}^c$. Let $\mathbf{X}^i = (X_1^i, X_2^i, ..., X_{|\mathcal{S}|}^i) : 0 \leq X_{j\in 1,2,..|\mathcal{S}|}^i \leq 1$; $\sum_{j=1}^{|\mathcal{S}|} X_j^i = 1 \ \forall i \in \{1,2,...,|\mathcal{A}|\}$ denote the random vector corresponding to the probabilities of reaching different states by taking an action $i \in A$. Let $\mathbf{Y}^{a_t}$ represent the sample collected at time step t by taking an action $a_t \in \mathcal{A}$. From Algorithm 1, the current estimate of state transition probabilities by taking an action $a \in \mathcal{A}$ would be $\overline{\mathbf{X}^a} = \frac{(\sum_{a_t : a_t = a} \mathbf{Y}^{a_t})}{|\{a_t : a_t = a\}|}$. Then for some $\delta \geq 0$,

$$P\left(\left\|\overline{\mathbf{X}^a} - \mathbb{E}[\overline{\mathbf{X}^a}]\right\|_1 > \delta\right) \leq \ P\left(\bigcup_{i=1}^{|S|} \left|\overline{X_i^a} - \mathbb{E}[\overline{X_i^a}]\right| > \frac{\delta}{|\mathcal{S}|}\right),$$

for some $i \in \{1,2,..,|\mathcal{S}|\}$, as whenever $\|\overline{\mathbf{X}^a} - \mathbb{E}[\overline{\mathbf{X}^a}]\|_1 > \delta$, $|\overline{X_i^a} - \mathbb{E}[\overline{X_i^a}]| \geq \frac{\delta}{|\mathcal{S}|}$ for at least one $i \in \{1,2,...,|\mathcal{S}|\}$. Further

$$P\left(\bigcup_{i=1}^{|\mathcal{S}|} \left|\overline{X_i^a} - \mathbb{E}[\overline{X_i^a}]\right| > \frac{\delta}{|\mathcal{S}|}\right) \leq \sum_{i=1}^{|S|} P\left(\left|\overline{X_i^a} - \mathbb{E}[\overline{X_i^a}]\right| > \frac{\delta}{|\mathcal{S}|}\right).$$

Using Hoeffding's inequality, which holds for all $i \in \{1,2,..,|\mathcal{S}|\}$,

$$P\left(\left\|\overline{\mathbf{X}^a} - \mathbb{E}[\overline{\mathbf{X}^a}]\right\|_1 > \delta\right) \leq |\mathcal{S}| e^{-\frac{2\lfloor T/|\mathcal{A}|\rfloor \delta^2}{|\mathcal{S}|}}. \qquad (11)$$

Since (12) is valid for any $a \in \mathcal{A}$, replacing $\|\overline{\mathbf{X}^a} - \mathbb{E}[\overline{\mathbf{X}^a}]\|_1$ with $\varepsilon = \max_{a \in A} \|\overline{\mathbf{X}^a} - \mathbb{E}[\overline{\mathbf{X}^a}]\|_1$ and substituting δ with $Q_D/\bar{R} \geq 0$ in (11), we have

$$P\left(\varepsilon > \frac{Q_D}{\bar{R}}\right) \leq |\mathcal{S}| e^{-\frac{2\lfloor T/|\mathcal{A}|\rfloor Q_D^2}{|S|\bar{R}^2}}.$$

From Theorem 1, the expression on the left is the probability of taking an unsafe action. Therefore, the probability of taking an unsafe action at any time step $t \geq T$ would be

$$P(s_{t+1} \in \mathcal{D} | s_t, \hat{\pi}^*(s_t)) = P\left(\varepsilon > \frac{Q_D}{\bar{R}}\right) \leq |\mathcal{S}| e^{-\frac{2\lfloor T/|\mathcal{A}|\rfloor Q_D^2}{|S|\bar{R}^2}}.$$

□

V. Numerical Experiments

In this section, we demonstrate the performance of the proposed exploration bonus with local side information while also validating the obtained theoretical guarantees on safety. In order to compare our results to [3], we consider the problem of the Mars 2020 Rover mission operating in Jezero crater [15], the same as the one presented therein. We assume that the sensors onboard the rover provide us with information on terrain features required for bounding the transition probability differences between states [16], [15]. Each of the states in our MDP belong to one of the three terrain types, *benign*, *rough*, and *rippled*, that differ in the (i) the similarity between transition probabilities of states within the same type, and (ii) the slip rate, defined as the probability of the rover moving in an unintended direction. For our simulations, we use the values of slip rate and difference between transition probabilities of states from [3] which are intended to be merely illustrative. The following five movements constitute the set of possible actions: up, down, left, right, and stay in the same position. Note that, for a given action, the actual movement of the rover could be different from the one intended due to slippage. Finally, we set the observation radius of the rover λ to 2.

A. Learning

This subsection compares the performance of direct and indirect weighted sampling algorithms for the sole objective of learning the environment dynamics by using the maximum learning error $e_{max} = \max_{s,a,s'} |\hat{\mathcal{P}}(s'|s,a) - \mathcal{P}(s'|s,a)|$ as the comparison metric. For the indirect sampling algorithm we use the weighting function, $w(\Delta) = (1-\Delta)^n$ where n is a tunable parameter, originally proposed in [3]. State-space was explored using the following policy: at any state s, choose an action a that is expected to result in a transition to a state s' that has been visited the least number of times. We ran the simulation for 10 million time steps with $n = 20$ and present the maximal learning errors in Fig. 1.

As expected, the indirect weighed sampling algorithms outperform direct sampling initially by converging very quickly, with the learning error in the case of local indirect sampling reaching a value of 0.09 within 7×10^5 time steps. However, we note that the error in indirect sampling algorithms converges to a non-zero value, due to the build-up of errors induced by reusing the samples. This is because the overall indirect samples collected at states using local side information are less than those collected by global indirect sampling. We resolve the issue of convergence to non-zero error by adding a time-varying component to the weighting function. Specifically, after every N time steps, we decrease the weights $\omega(s,s'), s \neq s'$, by a factor of K. This feature ensures that the samples collected in the long-term are more direct in nature. For our simulation, we took $N = 10^5$ and $K = 10$ and the time-varying weighting function behaves as expected, with its maximum error smaller than that of direct sampling in the initial time steps while converging to zero in long-term as shown in Fig. 1.

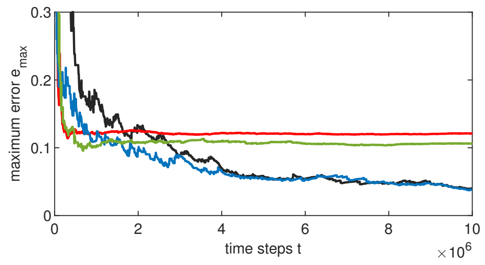

Fig. 1. The maximum learning errors for all the four versions of sampling: the black graph corresponds to the error in the direct sampling, the blue and green graphs represent the error in local indirect weighted sampling with time-varying and time-invariant weighting functions respectively, and the red graph corresponds to the error with global indirect sampling.

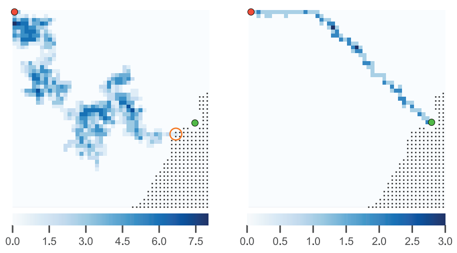

Fig. 2. Safety of the agent using BEB with direct sampling (left) and the proposed algorithm (right). Red and green circles denote initial and goal states, respectively. The orange circle shows agent reaching an unsafe state.

B. Control and Safety

We now compare BEB and the proposed algorithm in terms of safety and optimal control. We consider the optimal control problem of reaching a goal state s^* in the minimum amount of time and additionally assume that the rough terrain is not safe. The agent starts at a corner of the state space and is required to reach the goal state in the immediate neighborhood of rough terrain as shown in Fig. 2. We assign a reward of $(\|s^* - s\|)^{-1}$ for all states $s \in \mathcal{S} \backslash \{s^*\}$, a reward of 10 for the goal state s^* to ensure that the agent eventually reaches it, and a reward of 0 to all unsafe states to ensure that the agent avoids them. Policies were generated with the standard BEB algorithm and the proposed algorithm using the following values for the parameters – horizon length $H = 2$, $\beta = 2H^2$, $\gamma = 2H^2$, and $\mu = 2H$. These parameters, particularly the values of β and γ, are based on the values considered in [8]. From Fig. 2 we observe that the policy generated by the proposed algorithm leads the agent to safely reach the goal state located right next to unsafe states whereas the policy generated by BEB leads the agent to enter an unsafe state before reaching its goal state. We note that the simulation parameters were well outside the assumptions made for providing theoretical guarantees in the previous section. Hence, the safety guarantees obtained in Section IV are conservative; an agent using the proposed algorithm explores safely even for a non-zero horizon.

VI. Conclusions

We presented a framework that allows an agent to exploit local side information on similarities between the transition probabilities at different states in an MDP to accelerate learning by prioritizing states that are more valuable in terms of the total side information gain. Further, we provided guarantees on the safety of an agent using the proposed algorithm and validated all the proposed ideas using numerical experiments.

While this work develops the idea of indirect sampling by creating a more realistic framework, still much remains to be done in terms of providing theoretical guarantees on the learning speed and the error in the model learned using the proposed exploration bonus. Also, determining the exact relationship between tuning parameters in the benefit function and the control objective for a given environment is essential and was not addressed in our work. In addition to these open questions, extending the guarantees on safety to far more general scenarios would be a natural next step.

References

[1] P. Arena, P. Di Giamberardino, L. Fortuna, F. La Gala, S. Monaco, G. Muscato, A. Rizzo, and R. Ronchini, “Toward a mobile autonomous robotic system for Mars exploration,” *Planetary and Space Science*, vol. 52, no. 1-3, pp. 23–30, 2004.

[2] K. Yang, S. K. Gan, and S. Sukkarieh, “An efficient path planning and control algorithm for RUAVs in unknown and cluttered environments,” in *2nd International Symposium on UAVs*, 2009, pp. 101–122.

[3] M. Ornik, J. Fu, N. T. Lauffer, W. K. Perera, M. Alshiekh, M. Ono, and U. Topcu, “Expedited learning in MDPs with side information,” in *57th IEEE Conference on Decision and Control*. IEEE, 2018, pp. 1941–1948.

[4] S. Koenig, “Exploring unknown environments with real-time search or reinforcement learning,” in *Advances in Neural Information Processing Systems*, 1999, pp. 1003–1009.

[5] H. Gao, X. Song, L. Ding, K. Xia, N. Li, and Z. Deng, “Adaptive motion control of wheeled mobile robot with unknown slippage,” *International Journal of Control*, vol. 87, no. 8, pp. 1513–1522, 2014.

[6] A. L. Strehl, L. Li, and M. L. Littman, “Reinforcement learning in finite MDPs: PAC analysis,” *Journal of Machine Learning Research*, vol. 10, pp. 2413–2444, 2009.

[7] M. Kearns and S. Singh, “Near-optimal reinforcement learning in polynomial time,” *Machine Learning*, vol. 49, no. 2-3, pp. 209–232, 2002.

[8] J. Z. Kolter and A. Y. Ng, “Near-Bayesian exploration in polynomial time,” in *26th Annual International Conference on Machine Learning*. ACM, 2009, pp. 513–520.

[9] F. Berkenkamp, M. Turchetta, A. P. Schoellig, and A. Krause, “Safe model-based reinforcement learning with stability guarantees,” in *Advances in Neural Information Processing Systems*, 2017, pp. 908–918.

[10] T. M. Moldovan and P. Abbeel, “Safe exploration in Markov decision processes,” in *29th International Conference on Machine Learning*, 2012, pp. 1451–1458.

[11] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, “Safe reinforcement learning via shielding,” in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[12] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[13] R. I. Brafman and M. Tennenholtz, “R-MAX–a general polynomial time algorithm for near-optimal reinforcement learning,” *Journal of Machine Learning Research*, vol. 3, pp. 213–231, 2002.

[14] W. Hoeffding, “Probability inequalities for sums of bounded random variables,” *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 13–30, 1963.

[15] M. Ono, M. Heverly, B. Rothrock, E. Almeida, F. Calef, T. Soliman, N. Williams, H. Gengl, T. Ishimatsu, A. Nicholas, E. Stilley, K. Otsu, R. Lange, and S. M. Milkovich, “Mars 2020 site-specific mission performance analysis: Part 2. Surface traversability,” *2018 AIAA SPACE and Astronautics Forum and Exposition*, pp. 1–14, 2018.

[16] D. Helmick, A. Angelova, and L. Matthies, “Terrain adaptive navigation for planetary rovers,” *Journal of Field Robotics*, vol. 26, no. 4, pp. 391–410, 2009.