Improved Sample Complexity for Incremental Autonomous Exploration in MDPs

Jean Tarbouriech  
Facebook AI Research Paris & Inria Lille  
jean.tarbouriech@gmail.com

Matteo Pirotta  
Facebook AI Research Paris  
pirotta@fb.com

Michal Valko  
DeepMind Paris  
valkom@deepmind.com

Alessandro Lazaric  
Facebook AI Research Paris  
lazaric@fb.com

Abstract

We investigate the exploration of an unknown environment when no reward function is provided. Building on the incremental exploration setting introduced by Lim and Auer [1], we define the objective of learning the set of $\varepsilon$-optimal goal-conditioned policies attaining all states that are incrementally reachable within $L$ steps (in expectation) from a reference state $s_0$. In this paper, we introduce a novel model-based approach that interleaves discovering new states from $s_0$ and improving the accuracy of a model estimate that is used to compute goal-conditioned policies to reach newly discovered states. The resulting algorithm, DisCo, achieves a sample complexity scaling as $\tilde{O}(L^5 S_{L+\varepsilon}^L \Gamma_{L+\varepsilon} A \varepsilon^{-2})$, where $A$ is the number of actions, $S_{L+\varepsilon}$ is the number of states that are incrementally reachable from $s_0$ in $L + \varepsilon$ steps, and $\Gamma_{L+\varepsilon}$ is the branching factor of the dynamics over such states. This improves over the algorithm proposed in [1] in both $\varepsilon$ and $L$ at the cost of an extra $\Gamma_{L+\varepsilon}$ factor, which is small in most environments of interest. Furthermore, DisCo is the first algorithm that can return an $\varepsilon/c_{\text{min}}$-optimal policy for any cost-sensitive shortest-path problem defined on the $L$-reachable states with minimum cost $c_{\text{min}}$.

Finally, we report preliminary empirical results confirming our theoretical findings.

1 Introduction

In cases where the reward signal is not informative enough — e.g., too sparse, time-varying or even absent — a reinforcement learning (RL) agent needs to explore the environment driven by objectives other than reward maximization, see [e.g., 2, 3, 4, 5, 6]. This can be performed by designing intrinsic rewards to drive the learning process, for instance via state visitation counts [7, 8], novelty or prediction errors [9, 10, 11]. Other recent methods perform information-theoretic skill discovery to learn a set of diverse and task-agnostic behaviors [12, 13, 14]. Alternatively, goal-conditioned policies learned by carefully designing the sequence of goals during the learning process are often used to solve sparse reward problems [15] and a variety of goal-reaching tasks [16, 17, 18, 19].

While the approaches reviewed above effectively leverage deep RL techniques and are able to achieve impressive results in complex domains (e.g., Montezuma’s Revenge [15] or real-world robotic manipulation tasks [19]), they often lack substantial theoretical understanding and guarantees. Recently, some unsupervised RL objectives were analyzed rigorously. Some of them quantify how well the agent visits the states under a sought-after frequency, e.g., to induce a maximally entropic state distribution [20, 21, 22, 23]. While such strategies provably mimic their desired behavior via a Frank-Wolfe algorithmic scheme, they may not learn how to effectively reach any state of the environment and thus may not be sufficient to efficiently solve downstream tasks. Another relevant take is the reward-free RL paradigm of [24]: following its exploration phase, the agent is able to
compute a near-optimal policy for any reward function at test time. While this framework yields strong end-to-end guarantees, it is limited to the finite-horizon setting and the agent is thus unable to tackle tasks beyond finite-horizon, e.g., goal-conditioned tasks.

In this paper, we build on and refine the setting of incremental exploration of [1]: the agent starts at an initial state \( s_0 \) in an unknown, possibly large environment, and it is provided with a \texttt{RESET} action to restart at \( s_0 \). At a high level, in this setting the agent should explore the environment and stop when it has identified the \textit{tasks} within its reach and learned to \textit{master} each of them sufficiently well. More specifically, the objective of the agent is to learn a goal-conditioned policy for any state that can be reached from \( s_0 \) within \( L \) steps in expectation; such a state is said to be \( L \)-controllable. Lim and Auer [1] address this setting with the \texttt{UcbExplore} method for which they bound the number of exploration steps that are required to identify in an incremental way all \( L \)-controllable states (i.e., the algorithm needs to define a suitable stopping condition) and to return a set of policies that are able to reach each of them in \textit{at most} \( L + \varepsilon \) steps. A key aspect of \texttt{UcbExplore} is to first focus on simple states (i.e., states that can be reached within a few steps), learn policies to efficiently reach them, and leverage them to identify and tackle states that are increasingly more difficult to reach. This approach aims to avoid wasting exploration in the attempt of reaching states that are further than \( L \) steps from \( s_0 \) or that are too difficult to reach given the limited knowledge available at earlier stages of the exploration process. Our main contributions are:

- We strengthen the objective of incremental exploration and require the agent to learn \( \varepsilon \)-optimal goal-conditioned policies for any \( L \)-controllable state. Formally, let \( V^*(s) \) be the length of the shortest path from \( s_0 \) to \( s \), then the agent needs to learn a policy to navigate from \( s_0 \) to \( s \) in at most \( V^*(s) + \varepsilon \) steps, while in [1] any policy reaching \( s \) in \textit{at most} \( L + \varepsilon \) steps is acceptable.
- We design \texttt{DisCo}, a novel algorithm for incremental exploration. \texttt{DisCo} relies on an estimate of the transition model to compute goal-conditioned policies to the states observed so far and then use those policies to improve the accuracy of the model and incrementally discover new states.
- We derive a sample complexity bound for \texttt{DisCo} scaling as\(^1\) \( O\left(L^5 S L_{\varepsilon, \Gamma L_{\varepsilon}} A \varepsilon^{-2}\right) \), where \( A \) is the number of actions, \( S_{L+\varepsilon} \) is the number of states that are \textit{incrementally} controllable from \( s_0 \) in \( L + \varepsilon \) steps, and \( \Gamma L_{\varepsilon} \) is the branching factor of the dynamics over such incrementally controllable states. Not only is this sample complexity obtained for a more challenging objective than \texttt{UcbExplore}, but it also improves in both \( \varepsilon \) and \( L \) at the cost of an extra \( \Gamma L_{\varepsilon} \) factor, which is small in most environments of interest.
- Leveraging the model-based nature of \texttt{DisCo}, we can also readily compute an \( \varepsilon/c_{\min} \)-optimal policy for any cost-sensitive shortest-path problem defined on the \( L \)-controllable states with minimum cost \( c_{\min} \). This result serves as a goal-conditioned counterpart to the reward-free exploration framework defined by Jin et al. [24] for the finite-horizon setting.

2 Incremental Exploration to Discover and Control

In this section we expand [1], with a more challenging objective for autonomous exploration.

2.1 \( L \)-Controllable States

We consider a \textit{reward-free} Markov decision process [25, Sect. 8.3] \( M := (S, A, p, s_0) \). We assume a finite action space \( A \) with \( A = |A| \) actions, and a finite, possibly large state space \( S \) for which an upper bound \( S \) on its cardinality is known, i.e., \( |S| \leq S \). Each state-action pair \( (s, a) \in S \times A \) is characterized by an unknown transition probability distribution \( p(\cdot|s, a) \) over next states. We denote by \( \Gamma S := \max_{s \in S, a} ||p(s'|s, a)||_0 \) the largest branching factor of the dynamics over states in any subset \( S' \subseteq S \). The environment has no extrinsic reward, and \( s_0 \in S \) is a designated initial state. A deterministic stationary policy \( \pi : S \to A \) is a mapping between states to actions and we denote by \( \Pi \) the set of all possible policies. Since in environments with arbitrary dynamics the learner may get stuck in a state without being able to return to \( s_0 \), we introduce the following assumption.\(^2\)

\(^1\)We say that \( f(\varepsilon) = O(\varepsilon^a) \) if there are constants \( a, b \), such that \( f(\varepsilon) \leq a \cdot \varepsilon^b \log^b(\varepsilon) \).

\(^2\)Lim and Auer [1] originally considered a countable, possibly infinite state space; however this leads to a technical issue in the analysis of \texttt{UcbExplore} (acknowledged by the authors via personal communication and explained in App. E.3), which disappears by considering only finite state spaces.

\(^3\)This assumption should be contrasted with the finite-horizon setting, where each policy resets automatically after \( H \) steps, or assumptions on the MDP dynamics such as ergodicity or bounded diameter, which guarantee that it is always possible to find a policy navigating between any two states.

2
We measure the performance of a policy in navigating the MDP as follows. We illustrate the concept of controllable states in Fig. 1 for Assumption 1.

The objective of the learning agent is to Definition 2.

\( S \) incrementally controllable of \( S \) may traverse states that are not in \( S \) require a number of exploration steps that is exponential \( \pi \) is a policy \( \pi \) reaches the state in a number of steps less or equal than \( L \) states) and the notion of existence of a random realization where a state \( s \) is further the black states are not \( 0 \) than the black states, \( 0 \) does (i.e., \( s \) is in the vicinity of \( s_0 \)). We say that a state \( s \) is controlled if the agent can reliably navigate to it from \( s_0 \), that is, there exists an effective goal-conditioned policy — i.e., a shortest-path policy — from \( s_0 \) to \( s \).

Definition 3 (L-controllable states). Given a reference state \( s_0 \), we say that a state \( s \) is L-controllable if there exists a policy \( \pi \) such that \( v_\pi(s_0 \rightarrow s) \leq L \). The set of L-controllable states is then

\[
S_L := \{ s \in S : \min_{\pi \in \Pi} v_\pi(s_0 \rightarrow s) \leq L \}.
\]

We illustrate the concept of controllable states in Fig. 1 for \( L = 3 \). Interestingly, in the right figure, the black states are not L-controllable. In fact, there is no policy that can directly choose which one of the black states to reach. On the other hand, the red state, despite being in some sense further from \( s_0 \) than the black states, \( s_0 \) belongs to \( S_L \). In general, there is a crucial difference between the existence of a random realization where a state \( s \) is reached from \( s_0 \) in less than \( L \) steps (i.e., black states) and the notion of L-controllability, which means that there exists a policy that consistently reaches the state in a number of steps less or equal than \( L \) on average (i.e., red state). This explains the choice of the term controllable over reachable, since a state \( s \) is often said to be reachable if there is a policy \( \pi \) with a non-zero probability to eventually reach it, which is a weaker requirement.

Unfortunately, Lim and Auer [1] showed that in order to discover all the states in \( S_L \), the learner may require a number of exploration steps that is exponential in \( L \) or \( |S_L| \). Intuitively, this negative result is due to the fact that the minimum in Eq. 1 is over the set of all possible policies, including those that may traverse states that are not in \( S_L \). Hence, we similarly constrain the learner to focus on the set of incrementally controllable states.

Definition 4 (Incrementally controllable states \( S_L^+ \)). Let \( \prec \) be some partial order on \( S \). The set \( S_L^+ \) of states controllable in \( L \) steps w.r.t. \( \prec \) is defined inductively as follows. The initial state \( s_0 \)

\[\text{We refer the reader to [1, Sect. 2.1] for a more formal and complete characterization of this negative result.}\]
We say that an exploration algorithm solves the constraint problem \( \mathcal{L} \) restricted on \( S \) if for all \( s \in S \), there exists a policy \( \pi \) that satisfies the constraint \( v_{\pi}(s) \leq L \). The set \( S_{L}^{\omega} \) of incrementally \( L \)-controllable states is defined as \( S_{L}^{\omega} := \bigcup_{\omega} S_{L}^{\omega} \), where the union is over all possible partial orders.

By way of illustration, in Fig. 1 for \( L = 3 \), it holds that \( S_{L}^{+} = S_{L} \) in the left figure, whereas \( S_{L}^{+} = \{s_0\} \neq S_{L} \) in the right figure. Indeed, while the red state is \( L \)-controllable, it requires traversing the black states, which are not \( L \)-controllable.

### 2.2 AX Objectives

We are now ready to formalize two alternative objectives for Autonomous eXploration (AX) in MDPs.

**Definition 5 (AX sample complexity).** Fix any length \( L \geq 1 \), error threshold \( \varepsilon > 0 \) and confidence level \( \delta \in (0, 1) \). The sample complexities \( C_{AX} (\mathcal{A}, L, \varepsilon, \delta) \) and \( C_{AX}^{*} (\mathcal{A}, L, \varepsilon, \delta) \) are defined as the number of time steps required by a learning algorithm \( \mathcal{A} \) to identify a set \( K \supseteq S_{L}^{\omega} \) such that with probability at least \( 1 - \delta \), it has learned a set of policies \( \{\pi_s\}_{s \in K} \) that respectively verifies the following AX requirement:

\[
\begin{align*}
(AX_{L}) & \forall s \in K, v_{\pi_s}(s_0 \to s) \leq L + \varepsilon, \\
(AX^{*}) & \forall s \in K, v_{\pi_s}(s_0 \to s) \leq V_{S_{L}^{\omega}}(s_0 \to s) + \varepsilon.
\end{align*}
\]

Designing agents satisfying the objectives defined above introduces critical difficulties w.r.t. standard goal-directed learning in RL. First, the agent has to find accurate policies for a set of goals (i.e., all incrementally \( L \)-controllable states) and not just for one specific goal. On top of this, the set of desired goals itself (i.e., the set \( S_{L}^{\omega} \)) is unknown in advance and has to be estimated online. Specifically, \( AX_{L} \) is the original objective introduced in [1] and requires the agent to discover all the incrementally \( L \)-controllable states as fast as possible. At the end of the learning process, for each state \( s \in S_{L}^{\omega} \) the agent should return a policy that can reach \( s \) from \( s_0 \) in at most \( L \) steps (in expectation). Unfortunately, this may correspond to a rather poor performance in practice. Consider a state \( s \in S_{L}^{\omega} \) such that \( V_{S_{L}^{\omega}}(s_0 \to s) \ll L \), i.e., the shortest path between \( s_0 \) to \( s \) following policies restricted on \( S_{L}^{\omega} \) is much smaller than \( L \). Satisfying \( AX_{L} \) only guarantees that a policy reaching \( s \) in \( L \) steps is found. On the other hand, objective \( AX^{*} \) is more demanding, as it requires learning a near-optimal shortest-path policy for each state in \( S_{L}^{\omega} \). Since \( V_{S_{L}^{\omega}}(s_0 \to s) \leq L \) and the gap between the two quantities may be arbitrarily large, especially for states close to \( s_0 \) and far from the fringe of \( S_{L}^{\omega} \), \( AX^{*} \) is a significantly tighter objective than \( AX_{L} \) and it is thus preferable in practice.

We say that an exploration algorithm solves the AX problem if its sample complexity \( C_{AX} (\mathcal{A}, L, \varepsilon, \delta) \) in Def. 5 is polynomial in \( |K|, A, L, \varepsilon^{-1} \) and \( \log(S) \). Notice that requiring a logarithmic dependency on the size of \( S \) is crucial but nontrivial, since the overall state space may be large and we do not want the agent to waste time trying to reach states that are not \( L \)-controllable. The dependency on the (algorithmic-dependent and random) set \( K \) can be always replaced using the upper bound \( |K| \leq |S_{L}^{\omega} + \varepsilon| \), which is implied with high probability by both \( AX_{L} \) and \( AX^{*} \) conditions. Finally, notice that the error threshold \( \varepsilon > 0 \) has a two-fold impact on the performance of the algorithm. First, \( \varepsilon \) defines the largest set \( S_{L}^{\omega} \) that could be returned by the algorithm: the larger \( \varepsilon \), the bigger the set. Second, as \( \varepsilon \) increases, the quality (in terms of controllability and navigational precision) of the output policies worsens w.r.t. the shortest-path policy restricted on \( S_{L}^{\omega} \).

### 3 The DisCo Algorithm

The algorithm DisCo — for Discover and Control — is detailed in Alg. 1. It maintains a set \( K \) of “controllable” states and a set \( U \) of states that are considered “uncontrollable” so far. A state \( s \) is tagged as controllable when a policy to reach \( s \) in at most \( L + \varepsilon \) steps (in expectation from \( s_0 \)) has been found with high confidence, and we denote by \( \pi_s \) such policy. The states in \( U \) are states that have been discovered as potential members of \( S_{L}^{\omega} \), but the algorithm has yet to produce a policy to control any of them in less than \( L + \varepsilon \) steps. The algorithm stores an estimate of the transition model and it proceeds through rounds, which are indexed by \( k \) and incremented whenever a state in \( U \) gets transferred to the set \( K \), i.e., when the transition model reaches a level of accuracy sufficient
Algorithm 1: Algorithm DisCo

Input: Actions $A$, initial state $s_0$, confidence parameter $\delta \in (0, 1)$, error threshold $\varepsilon > 0$, $L \geq 1$ and (possibly adaptive) allocation function $\phi : P(S) \to \mathbb{N}$ (where $P(S)$ denotes the power set of $S$).

1. Initialize $k := 0$, $K_0 := \{s_0\}$, $U_0 := \{\}$ and a restricted policy $\pi_{s_0} \in \Pi(K_0)$.
2. Set $\varepsilon := \min\{\varepsilon, 1\}$ and continue := True.

while continue do

4. Set $k := k + 1$. //new round
5. // 1 Sample collection on $K$
6. For each $(s, a) \in K_k \times A$, execute policy $\pi_s$ until the total number of visits $N_k(s, a)$ to $(s, a)$ satisfies $N_k(s, a) \geq n_k := \phi(K_k)$. For each $(s, a) \in K_k \times A$, add $s' \sim p(s', a) \in U_k$ if $s' \notin K_k$.
7. // 2 Restriction of candidate states $U$
8. Compute transitions $\tilde{p}_k(s'|s, a)$ and $W_k := \{s' \in U_k : \exists (s, a) \in K_k \times A, \tilde{p}_k(s'|s, a) \geq \frac{1-\varepsilon/2}{L}\}.$
9. if $W_k$ is empty then
10. Set continue := False. //condition STOP1
11. else
12. // 3 Computation of the optimistic policies on $K$
13. for each state $s' \in W_k$ do
14. Compute $(\bar{u}_{s'}, \pi_{s'}) := \text{OVI}_{\text{SSP}}(K_k, A, s', N_k, \varepsilon)$, see Alg. 3 in App. D.1.
15. let $s^! := \arg \min_{s \in W_k} \bar{u}_{s}(s_0)$ and $\bar{u}^{!} := \bar{u}_{s^!}(s_0)$.
16. if $\bar{u}^{!} \geq L$ then
17. Set continue := False. //condition STOP2
18. else
19. // 4 State transfer from $U$ to $K$
20. Set $K_{k+1} := K_k \cup \{s^!\}$, $U_{k+1} := U_k \setminus \{s^!\}$ and $\pi_{s^!} := \pi_{s^!}$.

// 5 Policy consolidation: computation on the final set $K$
21. Set $K := k$.
22. for each state $s \in K_k$ do
23. Compute $(\bar{u}_s, \pi_s) := \text{OVI}_{\text{SSP}}(K_k, A, s, N_k, \varepsilon)$.

Output: the states $s$ in $K_K$ and their corresponding policy $\pi_s := \pi_s$.

to compute a policy to control one of the states encountered before. We denote by $K_k$ (resp. $U_k$) the set of controllable (resp. uncontrollable) states at the beginning of round $k$. DisCo stops at a round $K$ when it can confidently claim that all the remaining states outside of $K_K$ cannot be $L$-controllable.

At each round, the algorithm uses all samples observed so far to build an estimate of the transition model denoted by $\tilde{p}(s'|s, a) = N(s, a, s')/N(s, a)$, where $N(s, a)$ and $N(s, a, s')$ are counters for state-action and state-action-next state visitations. Each round is divided into two phases. The first is a sample collection phase. At the beginning of round $k$, the agent collects additional samples until $n_k := \phi(K_k)$ samples are available at each state-action pair in $K_k \times A$ (step 3). A key challenge lies in the careful (and adaptive) choice of the allocation function $\phi$, which we report in the statement of Thm. 1 (see Eq. 19 in App. D.4 for its exact definition). Importantly, the incremental construction of $K_k$ entails that sampling at each state $s \in K_k$ can be done efficiently. In fact, for all $s \in K_k$ the agent has already confidently learned a policy $\pi_s$ to reach $s$ in at most $L + \varepsilon$ steps on average (see how such policy is computed in the second phase). The generation of transitions $(s, a, s')$ for $(s, a) \in K_k \times A$ achieves two objectives at once. First, it serves as a discovery step, since all observed next states $s'$ not in $U_k$ are added to it — in particular this guarantees sufficient exploration at the fringe (or border) of the set $K_k$. Second, it improves the accuracy of the model $p$ in the states in $K_k$, which is essential in computing near-optimal policies and thus fulfilling the $AX^*$ condition.

The second phase does not require interacting with the environment and it focuses on the computation of optimistic policies. The agent begins by significantly restricting the set of candidate states in each round to alleviate the computational complexity of the algorithm. Namely, among all the states in $U_k$, it discards those that do not have a high probability of belonging to $S_L^{\bar{p}}$ by considering a restricted set $W_k \subset U_k$ (step 2). In fact, if the estimated probability $\tilde{p}_k$ of reaching a state $s \in U_k$ from any of the controllable states in $K_k$ is lower than $(1 - \varepsilon/2)/L$, then no shortest-path policy restricted on $K_k$ could get to $s$ from $s_0$ in less than $L + \varepsilon$ steps on average. Then for each state $s' \in W_k$, DisCo computes an optimistic policy restricted on $K_k$ to reach $s'$. Formally, for any candidate state $s' \in W_k$, we define the induced stochastic shortest path (SSP) MDP $M_k'$ with goal state $s'$ as follows.
We now present our main result: a sample complexity guarantee for DisCoAX.

Theorem 1. There exists an absolute constant (and thus on \(\tilde{\epsilon}\) Two cases are possible. If \(\tilde{\epsilon} \geq \frac{L + \epsilon}{\epsilon}\), since the accuracy of the model estimate on the state-action space \(K \times \mathcal{A}\) guarantees that the policy \(\hat{\pi}_s\) is able to reach the state \(s^*\) in less than \(L + \epsilon\) steps in expectation with high probability (i.e., \(s^*\) is incrementally \((L + \epsilon)\)-controllable). Otherwise, we can guarantee that \(S_L^{\star} \subseteq K\) with high probability. In the latter case, the algorithm terminates and, using the current estimates of the model, it recomputes an optimistic shortest-path policy \(\pi_s\) restricted on the final set \(K\) for each state \(s \in K\) (step 4). This policy consolidation step is essential to identify near-optimal policies restricted on the final set \(K\) (and thus on \(S_L^{\star}\)): indeed the expansion of the set of the so far controllable states may alter and refine the optimal goal-reaching policies restricted on it (see App. A).

Computational Complexity. Note that algorithmically, we do not need to define \(M'_1\) (Def. 6) over the whole state space \(S\) as we can limit it to \(K \cup \{s^\dagger\}\), i.e., the candidate state \(s^\dagger\) and the set \(K\) of so far controllable states. As shown in Thm. 1, this set can be significantly smaller than \(S\). In particular this implies that the computational complexity of the value iteration algorithm used to compute the optimistic policies is independent from \(S\) (see App. D for more details).

4 Sample Complexity Analysis of DisCoAX

We now present our main result: a sample complexity guarantee for DisCoAX for the AX* objective, which directly implies that \(AX_i\) is also satisfied.

Theorem 1. There exists an absolute constant \(\alpha > 0\) such that for any \(L \geq 1\), \(\epsilon \in (0, 1)\), and \(\delta \in (0, 1)\), if we set the allocation function \(\phi\) as

\[
\phi : \mathcal{X} \rightarrow \alpha \cdot \left( \frac{L \hat{\Theta}(\mathcal{X})}{\epsilon^2} \log \left( \frac{L \mathcal{S} \mathcal{A}}{\epsilon \delta} \right) + \frac{L^2 |\mathcal{X}|}{\epsilon} \log \left( \frac{L \mathcal{S} \mathcal{A}}{\epsilon \delta} \right) \right),
\]

with \(\hat{\Theta}(\mathcal{X}) := \max_{(s, a) \in \mathcal{X} \times \mathcal{A}} \left( \sum_{s' \in \mathcal{X}} \sqrt{p(s'|s, a)(1 - p(s'|s, a))} \right)^2\), then the algorithm DisCoAX (Alg. 1) satisfies the following sample complexity bound for AX*

\[
C_{AX^\star}(\text{DisCoAX}, L, \epsilon, \delta) = \tilde{O} \left( \frac{L^5 \Gamma_{L+\epsilon} S_{L+\epsilon} A}{\epsilon^2} + \frac{L^3 S_{L+\epsilon}^2 A}{\epsilon} \right),
\]

where \(S_{L+\epsilon} := |S_{L+\epsilon}^\star|\) and

\[
\Gamma_{L+\epsilon} := \max_{(s, a) \in S_{L+\epsilon}^\star \times \mathcal{A}} \|p(s'|s, a)\|_{0} \leq S_{L+\epsilon}
\]

is the maximal support of the transition probabilities \(p(\cdot|s, a)\) restricted to the set \(S_{L+\epsilon}^\star\).

Given the definition of AX*, Thm. 1 implies that DisCoAX 1) terminates after \(C_{AX^\star}(\text{DisCoAX}, L, \epsilon, \delta)\) time steps, 2) discovers a set of states \(K \supseteq S_L^\star + \epsilon\) with \(|K| \leq S_{L+\epsilon}\), and for each \(s \in K\) outputs a policy \(\pi_s\), which is \(\epsilon\)-optimal w.r.t. policies restricted on \(S_L^\star\), i.e., \(\nu_{\pi_s}(s_0 \rightarrow s) \leq V_{S_L^\star}^*(s_0 \rightarrow s) + \epsilon\). Note that Eq. 3 displays only a logarithmic dependency on \(S\), the total number of states. This property on the sample complexity of DisCoAX, along with its \(S\)-independent computational complexity, is significant when the state space \(S\) grows large w.r.t. the unknown set of interest \(S_L^\star\).

\[\text{In words, all actions at states in } K \text{ behave exactly as in } M \text{ and suffer a unit cost, in all states outside } K \text{ only the reset action to } s_0 \text{ is available with a unit cost, and all actions at the goal } s' \text{ induce a zero-cost self-loop.}\]
4.1 Proof Sketch of Theorem 1

While the complete proof is reported in App. D, we now provide the main intuition behind the result.

**State Transfer from $\mathcal{U}$ to $\mathcal{K}$ (step 4).** Let us focus on a round $k$ and a state $s^\dagger \in \mathcal{U}_k$ that gets added to $\mathcal{K}_k$. For clarity we remove in the notation the round $k$, goal state $s^\dagger$ and starting state $s_0$. We denote by $v$ and $\tilde{v}$ the value functions of the candidate policy $\bar{\pi}$ in the true and optimistic model respectively, and by $\bar{u}$ the quantity w.r.t. which $\bar{\pi}$ is optimistically greedy. We aim to prove that $s^\dagger \in S_{L+\varepsilon}$ (with high probability). The main chain of inequalities underpinning the argument is

$$v \leq |v - \tilde{v}| + \tilde{v} \leq \frac{\varepsilon}{2} + \frac{\varepsilon}{2} + \bar{u} + \frac{\varepsilon}{2} \leq L + \varepsilon,$$

where (c) is guaranteed by algorithmic construction and (b) stems from the chosen level of value iteration accuracy. Inequality (a) has the flavor of a simulation lemma for SSP, by relating the shortest-path value function of a same policy between two models (the true one and the optimistic one). Importantly, when restricted to $\mathcal{K}$ these two models are close in virtue of the algorithmic design which enforces the collection of a minimum amount of samples at each state-action pair of $\mathcal{K} \times \mathcal{A}$, denoted by $n$. Specifically, we obtain that

$$|v - \tilde{v}| = \mathcal{O}\left(\sqrt{\frac{L^2|\mathcal{K}|}{n} + \frac{L^2}{n}}\right), \quad \text{with} \quad \Gamma_{\mathcal{K}} := \max_{(s,a) \in \mathcal{K} \times \mathcal{A}} \left\{\max_{s' \in \mathcal{K}} \|p(s'|s,a)\|_\mathcal{A} \leq |\mathcal{K}|\right\}.$$

Note that $\Gamma_{\mathcal{K}}$ is the branching factor restricted to the set $\mathcal{K}$. Our choice of $n$ (given in Eq. 2) is then dictated to upper bound the above quantity by $\varepsilon/2$ in order to satisfy inequality (a). Let us point out that, interestingly yet unfortunately, the structure of the problem does not appear to allow for technical variance-aware improvements seeking to lower the value of $n$ prescribed above (indeed the AX framework requires to analytically encompass the uncontrollable states $\mathcal{U}$ into a single meta state with higher transitional uncertainty, see App. D for details).

**Termination of the Algorithm.** Since $S_{\mathcal{L}}^+$ is unknown, we have to ensure that none of the states in $S_{\mathcal{L}}^+$ are “missed”. As such, we prove that with overwhelming probability, we have $S_{\mathcal{L}}^+ \subseteq \mathcal{K}_K$ when the algorithm terminates at a round denoted by $K$. There remains to justify the final near-optimal guarantee w.r.t. the set of policies $\Pi(S_{\mathcal{L}}^+)$. Leveraging that step 5 recomputes the policies $(\pi_x)_{s \in \mathcal{K}_K}$ on the final set $\mathcal{K}_K$, we establish the following chain of inequalities

$$v \leq |v - \tilde{v}| + \tilde{v} \leq \frac{\varepsilon}{2} + \frac{\varepsilon}{2} + \bar{u} + \frac{\varepsilon}{2} \leq V_{\mathcal{K}_K}^\ast + \varepsilon \leq V_{S_{\mathcal{L}}^+}^\ast + \varepsilon,$$

where (a) and (b) are as in Eq. 4, (c) leverages optimism and (d) stems from the inclusion $S_{\mathcal{L}}^+ \subseteq \mathcal{K}_K$.

**Sample Complexity Bound.** The choice of allocation function $\phi$ in Eq. 2 bounds $n_K$ which is the total number of samples required at each state-action pair in $\mathcal{K}_K \times \mathcal{A}$. We then compute a high-probability bound $\psi$ on the time steps needed to collect a given sample, and show that it scales as $\mathcal{O}(L)$. Since the sample complexity is solely induced by the sample collection phase (step 3), it can be bounded by the quantity $\psi n_K |\mathcal{K}_K| |\mathcal{A}|$. Putting everything together yields the bound of Thm. 1.

4.2 Comparison with UcbExplore [1]

We start recalling the critical distinction that DisCo succeeds in tackling problem AX*, while UcbExplore [1] fails to do so (see App. A for details on the AX objectives). Nonetheless, in the following we show that even if we restrict our attention to AXL, for which UcbExplore is designed, DisCo yields a better sample complexity in most of the cases. From [1], UcbExplore verifies

$$C_{AXL}(\text{UcbExplore}, L, \varepsilon, \delta) = \mathcal{O}\left(\frac{L^6 S_{L+\varepsilon} A}{\varepsilon^3}\right).$$

Eq. 6 shows that the sample complexity of UcbExplore is linear in $S_{L+\varepsilon}$, while for DisCo the dependency is somewhat worse. In the main-order term $\mathcal{O}(1/\varepsilon^2)$ of Eq. 3, the bound depends linearly on $S_{L+\varepsilon}$ but also grows with the branching factor $\Gamma_{L+\varepsilon}$, which is not the “global” branching factor.

---

Note that if we replace the error of $\varepsilon$ for AXL with an error of $L \varepsilon$ as in [1], we recover the sample complexity of $\mathcal{O}(L^3 S_{L+\varepsilon} A/\varepsilon^3)$ stated in [1, Thm. 8].
but denotes the number of possible next states in $S_L^{\rightarrow\varepsilon}$ starting from $S_L^{\rightarrow\varepsilon}$. While in general we only have $\Gamma_{L+\varepsilon} \leq S_{L+\varepsilon}$, in many practical domains (e.g., robotics, user modeling), each state can only transition to a small number of states, i.e., we often have $\Gamma_{L+\varepsilon} = O(1)$ as long as the dynamics is not too “chaotic”. While $\text{DisCo}$ does suffer from a quadratic dependency on $S_{L+\varepsilon}$ in the second term of order $O(1/\varepsilon)$, we notice that for any $S_{L+\varepsilon} \leq L^3\varepsilon^{-2}$ the bound of $\text{DisCo}$ is still preferable. Furthermore, since for $\varepsilon \to 0$, $S_{L+\varepsilon}$ tends to $S_L$, the condition is always verified for small enough $\varepsilon$.

Compared to $\text{DisCo}$, the sample complexity of $\text{UcbExplore}$ is worse in both $\varepsilon$ and $L$. As stressed in Sect. 2.2, the better dependency on $\varepsilon$ both improves the quality of the output goal-reaching policies as well as reduces the number of incrementally $(L + \varepsilon)$-controllable states returned by the algorithm. It is interesting to investigate why the bound of [1] (Eq. 6) inherits a $O(\varepsilon^{-3})$ dependency. As reviewed in App. E, $\text{UcbExplore}$ alternates between two phases of state discovery and policy evaluation. The optimistic policies computed by $\text{UcbExplore}$ solve a finite-horizon problem (with horizon set to $H_{\text{Ucb}}$). However, minimizing the expected time to reach a target state is intrinsically an SSP problem, which is exactly what $\text{DisCo}$ leverages. By computing policies that solve a finite-horizon problem (note that $\text{UcbExplore}$ resets every $H_{\text{Ucb}}$ time steps), [1] sets the horizon to $H_{\text{Ucb}} := [L + L^2\varepsilon^{-1}]$, which leads to a policy-evaluation phase with sample complexity scaling as $O(H_{\text{Ucb}}\varepsilon^{-2}) = O(\varepsilon^{-3})$. Since the rollout budget of $O(\varepsilon^{-3})$ is hard-coded into the algorithm, the dependency on $\varepsilon$ of $\text{UcbExplore}$’s sample complexity cannot be improved by a more refined analysis; instead a different algorithmic approach is required such as the one employed by $\text{DisCo}$.

### 4.3 Goal-Free Cost-Free Exploration on $S_L^{\rightarrow\varepsilon}$ with $\text{DisCo}$

A compelling advantage of $\text{DisCo}$ is that it achieves an accurate estimation of the environment’s dynamics restricted to the unknown subset of interest $S_L^{\rightarrow\varepsilon}$. In contrast to $\text{UcbExplore}$ which needs to restart its sample collection from scratch whenever $L, \varepsilon$ or some transition costs change, $\text{DisCo}$ can thus be robust to changes in such problem parameters. At the end of its exploration phase in Alg. 1, $\text{DisCo}$ is able to perform zero-shot planning to solve other tasks restricted on $S_L^{\rightarrow\varepsilon}$, such as cost-sensitive ones. Indeed in the following we show how the $\text{DisCo}$ agent is able to compute an $\varepsilon/c_{\min}$-optimal policy for any stochastic shortest-path problem on $S_L^{\rightarrow\varepsilon}$ with goal state $s \in S_L^{\rightarrow\varepsilon}$ (i.e., $s$ is absorbing and zero-cost) and cost function lower bounded by $c_{\min} > 0$.

**Corollary 1.** There exists an absolute constant $\beta > 0$ such that for any $L \geq 1$, $\varepsilon \in (0, 1]$ and $c_{\min} \in (0, 1]$ verifying $\varepsilon \leq \beta \cdot (L c_{\min})$, with probability at least $1 - \delta$, for whatever goal state $s \in S_L^{\rightarrow\varepsilon}$ and whatever cost function $c \in [c_{\min}, 1]$, $\text{DisCo}$ can compute (after its exploration phase, without additional environment interaction) a policy $\pi_{s,c}$ whose SSP value function $V_{\pi_{s,c}}$ verifies

$$V_{\pi_{s,c}}(s_0 \rightarrow s) \leq V_{S_L^{\rightarrow\varepsilon}}^*(s_0 \rightarrow s) + \frac{\varepsilon}{c_{\min}},$$

where $V_{\pi}(s_0 \rightarrow s) := E\left[\sum_{t=1}^{\tau_{s,s_0}} c(s_t, \pi(s_t)) \mid s_1 = s_0\right]$ is the SSP value function of a policy $\pi$ and $V_{S_L^{\rightarrow\varepsilon}}^*(s_0 \rightarrow s) := \min_{\pi \in \Pi(S_L^{\rightarrow\varepsilon})} V_{\pi}(s_0 \rightarrow s)$ is the optimal SSP value function restricted on $S_L^{\rightarrow\varepsilon}$.

It is interesting to compare Cor. 1 with the reward-free exploration framework recently introduced by Jin et al. [24] in finite-horizon. At a high level, the result in Cor. 1 can be seen as a counterpart of [24] beyond finite-horizon problems, specifically in the goal-conditioned setting. While the parameter $L$ defines the horizon of interest for $\text{DisCo}$, resetting after every $L$ steps (as in finite-horizon) would prevent the agent to identify $L$-controllable states and lead to poor performance. This explains the distinct technical tools used: while [24] executes finite-horizon no-regret algorithms, $\text{DisCo}$ deploys SSP policies restricted on the set of states that it “controls” so far. Algorithmically, both approaches seek to build accurate estimates of the transitions on a specific (unknown) state space of interest: the so-called “significant” states within $H$ steps for [24], and the incrementally $L$-controllable states $S_L^{\rightarrow\varepsilon}$ for $\text{DisCo}$. Bound-wise, the cost-sensitive $A^*$ problem inherits the critical role of the minimum cost $c_{\min}$ in SSP problems (see App. C and e.g., [27, 28, 29]), which is reflected in the accuracy of Cor. 1 scaling inversely with $c_{\min}$. Another interesting element of comparison is the dependency on the size of the state space. While the algorithm introduced in [24] is robust w.r.t. states that can be reached with very low probability, it still displays a polynomial dependency on the total number of states $S$. On the other hand, $\text{DisCo}$ has only a logarithmic dependency on $S$, while it directly depends on the number of $(L + \varepsilon)$-controllable states, which shows that $\text{DisCo}$ effectively adapts to the state space of interest and it ignores all other states. This result is significant since not only $S_{L+\varepsilon}$ can be arbitrarily smaller than $S$, but also because the set $S_L^{\rightarrow\varepsilon}$ itself is initially unknown to the algorithm.
We compare the sample complexity empirically achieved by DiSCO and UcbExplore. Fig. 2 depicts the time needed to identify all the incrementally $L$-controllable states when $L = 4.5$ for different values of $\varepsilon$, on a confusing chain domain. Note that the sample complexity is achieved soon after, when the algorithm can confidently discard all the remaining states as non-controllable (it is reported in Tab. 2 of App. F). We observe that DiSCO outperforms UcbExplore for any value of $\varepsilon$. In particular, the gap in performance increases as $\varepsilon$ decreases, which matches the theoretical improvement in sample complexity from $O(\varepsilon^{-3})$ for UcbExplore to $\tilde{O}(\varepsilon^{-2})$ for DiSCO. On a second environment — the combination lock problem introduced in [31] — we notice that DiSCO again outperforms UcbExplore, as shown in App. F.

Another important feature of DiSCO is that it targets the tighter objective $AX^*$, whereas UcbExplore is only able to fulfill objective $AX_1$ and may therefore elect suboptimal policies. In App. F we show empirically that, as expected theoretically, this directly translates into higher-quality goal-reaching policies recovered by DiSCO.

5 Numerical Simulation

In this section, we provide the first evaluation of algorithms in the incremental autonomous exploration setting. In the implementation of both DiSCO and UcbExplore, we remove the logarithmic and constant terms for simplicity. We also boost the empirical performance of UcbExplore in various ways, for example by considering confidence intervals derived from the empirical Bernstein inequality (see [30]) as opposed to Hoeffding as done in [1]. We refer the reader to App. F for details on the algorithmic configurations and on the environments considered.

We compare the sample complexity empirically achieved by DiSCO and UcbExplore. Fig. 2 depicts the time needed to identify all the incrementally $L$-controllable states when $L = 4.5$ for different values of $\varepsilon$, on a confusing chain domain. Note that the sample complexity is achieved soon after, when the algorithm can confidently discard all the remaining states as non-controllable (it is reported in Tab. 2 of App. F). We observe that DiSCO outperforms UcbExplore for any value of $\varepsilon$. In particular, the gap in performance increases as $\varepsilon$ decreases, which matches the theoretical improvement in sample complexity from $O(\varepsilon^{-3})$ for UcbExplore to $\tilde{O}(\varepsilon^{-2})$ for DiSCO. On a second environment — the combination lock problem introduced in [31] — we notice that DiSCO again outperforms UcbExplore, as shown in App. F.

Another important feature of DiSCO is that it targets the tighter objective $AX^*$, whereas UcbExplore is only able to fulfill objective $AX_1$ and may therefore elect suboptimal policies. In App. F we show empirically that, as expected theoretically, this directly translates into higher-quality goal-reaching policies recovered by DiSCO.

6 Conclusion and Extensions

Connections to existing deep-RL methods. While we primarily focus the analysis of DiSCO in the tabular case, we believe that the formal definition of $AX$ problems and the general structure of DiSCO may also serve as a theoretical grounding of many recent approaches to unsupervised exploration. For instance, it is interesting to draw a parallel between DiSCO and the ideas behind Go-Explore [32]. Go-Explore similarly exploits the following principles: (1) remember states that have previously been visited, (2) first return to a promising state (without exploration), (3) then explore from it. Go-Explore assumes that the world is deterministic and resettable, meaning that one can reset the state of the simulator to a previous visit to that cell. Very recently [15], the same authors proposed a way to relax this requirement by training goal-conditioned policies to reliably return to cells in the archive during the exploration phase. In this paper, we investigated the theoretical dimension of this direction, by provably learning such goal-conditioned policies for the set of incrementally controllable states.

Future work. Interesting directions for future investigation include: 1) Deriving a lower bound for the $AX$ problems; 2) Integrating DiSCO into the meta-algorithm MNM [33] which deals with incremental exploration for $AX_1$ in non-stationary environments; 3) Extending the problem to continuous state space and function approximation; 4) Relaxing the definition of incrementally controllable states and relaxing the performance definition towards allowing the agent to have a non-zero but limited sample complexity of learning a shortest-path policy for any state at test time.
Broader Impact

This paper makes contributions to the fundamentals of online learning (RL) and due to its theoretical nature, we see no ethical or immediate societal consequence of our work.

References

[1] Shiau Hong Lim and Peter Auer. Autonomous exploration for navigating in MDPs. In Conference on Learning Theory, pages 40–1, 2012.

[2] Jürgen Schmidhuber. A possibility for implementing curiosity and boredom in model-building neural controllers. In Proc. of the international conference on simulation of adaptive behavior: From animals to animats, pages 222–227, 1991.

[3] Nuttapong Chentanez, Andrew G Barto, and Satinder P Singh. Intrinsically motivated reinforcement learning. In Advances in neural information processing systems, pages 1281–1288, 2005.

[4] Pierre-Yves Oudeyer and Frederic Kaplan. What is intrinsic motivation? a typology of computational approaches. Frontiers in neurorobotics, 1:6, 2009.

[5] Satinder Singh, Richard L Lewis, Andrew G Barto, and Jonathan Sorg. Intrinsically motivated reinforcement learning: An evolutionary perspective. IEEE Transactions on Autonomous Mental Development, 2(2):70–82, 2010.

[6] Adrien Baranes and Pierre-Yves Oudeyer. Intrinsically motivated goal exploration for active motor learning in robots: A case study. In 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 1766–1773. IEEE, 2010.

[7] Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. In Advances in neural information processing systems, pages 1471–1479, 2016.

[8] Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, Xi Chen, Yan Duan, John Schulman, Filip DeTurck, and Pieter Abbeel. # exploration: A study of count-based exploration for deep reinforcement learning. In Advances in neural information processing systems, pages 2753–2762, 2017.

[9] Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Variational information maximizing exploration. Advances in Neural Information Processing Systems (NIPS), 2016.

[10] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pages 16–17, 2017.

[11] Mohammad Gheshlaghi Azar, Bilal Piot, Bernardo Avila Pires, Jean-Bastian Grill, Florent Altché, and Rémi Munos. World discovery models. arXiv preprint arXiv:1902.07685, 2019.

[12] Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In International Conference on Learning Representations, 2019.

[13] Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-aware unsupervised discovery of skills. In International Conference on Learning Representations, 2020.

[14] Víctor Campos Camúñez, Alex Trott, Caiming Xiong, Richard Socher, Xavier Giró Nieto, and Jordi Torres Viñals. Explore, discover and learn: unsupervised discovery of state-covering skills. In International Conference on Machine Learning, pages 1317–1327. PMLR, 2020.

[15] Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O Stanley, and Jeff Clune. First return then explore. arXiv preprint arXiv:2004.12919, 2020.
[16] Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. Automatic goal generation for reinforcement learning agents. In International Conference on Machine Learning, pages 1515–1528, 2018.

[17] Cédric Colas, Pierre Fournier, Mohamed Chetouani, Olivier Sigaud, and Pierre-Yves Oudeyer. Curious: intrinsically motivated modular multi-goal reinforcement learning. In International Conference on Machine Learning, pages 1331–1340. PMLR, 2019.

[18] David Warde-Farley, Tom Van de Wiele, Tejas Kulkarni, Catalin Ionescu, Steven Hansen, and Volodymyr Mnih. Unsupervised control through non-parametric discriminative rewards. In International Conference on Learning Representations, 2019.

[19] Vitchyr H Pong, Murat Z Bal, Steven Lin, Ashvin Nair, Shikhar Bahl, and Sergey Levine. Skew-fit: State-covering self-supervised reinforcement learning. In International Conference on Machine Learning, pages 7783–7792. PMLR, 2020.

[20] Elad Hazan, Sham Kakade, Karan Singh, and Abby Van Soest. Provably efficient maximum entropy exploration. In International Conference on Machine Learning, pages 2681–2691, 2019.

[21] Jean Tarbouriech and Alessandro Lazaric. Active exploration in markov decision processes. In The 22nd International Conference on Artificial Intelligence and Statistics, pages 974–982, 2019.

[22] Wang Chi Cheung. Exploration-exploitation trade-off in reinforcement learning on online markov decision processes with global concave rewards. arXiv preprint arXiv:1905.06466, 2019.

[23] Jean Tarbouriech, Shubhanshu Shekhar, Matteo Pirotta, Mohammad Ghavamzadeh, and Alessandro Lazaric. Active model estimation in markov decision processes. In Conference on Uncertainty in Artificial Intelligence, 2020.

[24] Chi Jin, Akshay Krishnamurthy, Max Simchowitz, and Tiancheng Yu. Reward-free exploration for reinforcement learning. In International Conference on Machine Learning, pages 4870–4879. PMLR, 2020.

[25] Martin L Puterman. *Markov Decision Processes.: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2014.

[26] Dimitri Bertsekas. *Dynamic programming and optimal control*, volume 2. 2012.

[27] Jean Tarbouriech, Evrard Garcelon, Michal Valko, Matteo Pirotta, and Alessandro Lazaric. No-regret exploration in goal-oriented reinforcement learning. In International Conference on Machine Learning, pages 9428–9437. PMLR, 2020.

[28] Aviv Rosenberg, Alon Cohen, Yishay Mansour, and Haim Kaplan. Near-optimal regret bounds for stochastic shortest path. In International Conference on Machine Learning, pages 8210–8219. PMLR, 2020.

[29] Dimitri P Bertsekas and Huizhen Yu. Stochastic shortest path problems under weak conditions. Lab. for Information and Decision Systems Report LIDS-P-2909, MIT, 2013.

[30] Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. In Proceedings of the 34th International Conference on Machine Learning-Volume 70, pages 263–272. JMLR. org, 2017.

[31] Mohammad Gheshlaghi Azar, Vicenc Gómez, and Hilbert J Kappen. Dynamic policy programming. Journal of Machine Learning Research, 13(Nov):3207–3245, 2012.

[32] Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O Stanley, and Jeff Clune. Go-explore: a new approach for hard-exploration problems. arXiv preprint arXiv:1901.10995, 2019.

[33] Pratik Gajane, Ronald Ortner, Peter Auer, and Csaba Szepesvari. Autonomous exploration for navigating in non-stationary CMPs. arXiv preprint arXiv:1910.08446, 2019.
[34] Blai Bonet. On the speed of convergence of value iteration on stochastic shortest-path problems. *Mathematics of Operations Research*, 32(2):365–373, 2007.

[35] Jean-Yves Audibert, Rémi Munos, and Csaba Szepesvári. Tuning bandit algorithms in stochastic environments. In *International conference on algorithmic learning theory*, pages 150–165. Springer, 2007.

[36] Andreas Maurer and Massimiliano Pontil. Empirical bernstein bounds and sample variance penalization. *arXiv preprint arXiv:0907.3740*, 2009.

[37] Dimitri P Bertsekas and John N Tsitsiklis. An analysis of stochastic shortest path problems. *Mathematics of Operations Research*, 16(3):580–595, 1991.

[38] Ronan Fruit, Matteo Pirotta, and Alessandro Lazaric. Improved analysis of ucrl2 with empirical bernstein inequality. *arXiv preprint arXiv:2007.05456*, 2020.

[39] Abbas Kazerouni, Mohammad Ghavamzadeh, Yasin Abbasi, and Benjamin Van Roy. Conservative contextual linear bandits. In *Advances in Neural Information Processing Systems*, pages 3910–3919, 2017.


Appendix

A Autonomous Exploration Objectives

We recall the two AX objectives stated in Def. 5: for any length \( L \geq 1 \), error threshold \( \varepsilon > 0 \) and confidence level \( \delta \in (0, 1) \), the sample complexities \( C_{AX_1}(\mathcal{X}, L, \varepsilon, \delta) \) and \( C_{AX^*}(\mathcal{X}, L, \varepsilon, \delta) \) are defined as the number of time steps required by a learning algorithm \( \mathcal{A} \) to identify a set \( K \supseteq S^+_L \) such that with probability at least \( 1 - \delta \), it has learned a set of policies \( \{ \pi_s \}_{s \in K} \) that respectively verifies the following AX requirement

\[
\begin{align*}
(AX_L) & \quad \forall s \in K, v_{\pi_s}(s_0 \rightarrow s) \leq L + \varepsilon, \\
(AX^*) & \quad \forall s \in K, v_{\pi_s}(s_0 \rightarrow s) \leq V^*_{S^+_L}(s_0 \rightarrow s) + \varepsilon.
\end{align*}
\]

As we explain in Sect. 4, DisCo (Alg. 1) succeeds in tackling condition \( AX^* \), whereas UcbExplore [1], which is designed to tackle condition \( AX_1 \), is unable to tackle \( AX^* \). Note that the algorithmic design of UcbExplore entails that it computes policies whose value function implicitly targets \( V^*_K \), with \( K \) the current set of controllable states. While \( V^*_K \) is always smaller than \( L \), UcbExplore cannot provide any tightness guarantees w.r.t. \( V^*_K \) since it has no guarantee that the transition dynamics are estimated well enough on \( K \). An additional challenge with which UcbExplore fails to cope is the fact that the set \( K \) increases over time and thus unlocks new states and paths, which may be useful to improve its shortest-path policies for previously discovered states.

To better understand this phenomenon, let us introduce an alternative condition \( AX' \)—tighter than \( AX_1 \), but looser than \( AX^* \)—which stems from the challenge of not knowing \( S^+_L \) in advance. We define \( AX' \) as follows: for any state \( s \in S^+_L \), the objective is to find a policy that can reach \( s \) from \( s_0 \) in at most \( L' + \varepsilon \) steps on average, where \( L' := \min\{l \leq L : s \in S^+_l\} \), i.e.,

\[
(AX') \quad \forall s \in K, v_{\pi_s}(s_0 \rightarrow s) \leq L' + \varepsilon, \text{ where } L' := \min\{l \leq L : s \in S^+_l\}.
\]

As mentioned in [1, Corollary 9], it is possible to run separate instances of UcbExplore with increasing \( L_n = 1 + n \varepsilon \) from \( n = 0 \) to \( \lceil \frac{L_n - 1}{\varepsilon} \rceil \) (i.e., until \( n \) satisfies \( L_{n-1} \leq L \leq L_n \)). This verifies the condition \( AX' \) at the cost of a worsened dependency on both \( \varepsilon \) and \( L \) as follows

\[
C_{AX'}(\text{UcbExplore}, L, \varepsilon, \delta) = O\left(\frac{L^2 S_L l A}{\varepsilon^4}\right).
\]

While \( AX' \) is tighter than \( AX_1 \), it may be arbitrarily loose compared to \( AX^* \), which illustrates the intrinsic limitations in UcbExplore design. UcbExplore incrementally expands a set of “controllable” states \( K \): starting with \( K_0 = \{s_0\} \), at time \( t \) a state \( s \) is added to \( K_t \) whenever UcbExplore can confidently assess that it managed to learn a policy reaching \( s \) in less than \( L \) steps. Since at time \( t \) UcbExplore can only consider policies restricted to the controllable states \( K_t \), even the shortest-path policy computed to reach \( s \) at time \( t \) may not be \( \varepsilon \)-optimal w.r.t. the whole set \( S^+_L \). Indeed, every time a state is added to \( K \), this state may unlock new paths which may, for previously controllable states, allow for better shortest-path policies restricted on the updated \( K \). Fig. 3 illustrates this behavior, where the state \( y \) unlocks a fast path from \( y \) to \( x \) which should be taken in \( y \) instead of resetting to \( s_0 \). Consequently, if the agent seeks to tackle condition \( AX^* \), it must have the faculty to backtrack, i.e., continuously update both its belief of the vicinity \( (K) \) and its notion of optimality on the vicinity \( V^*_K \). Unfortunately, UcbExplore can only compute policies targeting \( V^*_K \) with \( K \) the current set of controllable states, but it fails to be accurate enough to revise such policies as the set of

![Figure 3: Let \( X := \{s_0\} \cup \{x\} \) and \( Y := X \cup \{y\} \). For any \( l \geq 1 \), suppose that from \( s_0 \), the agent reaches \( x \) in \( l \) steps with probability 1/2, or reaches \( y \) in \( l + 1 \) steps with probability 1/2. If the goal state is \( x \), constraining an agent to use policies restricted to \( X \) (i.e., that reset to \( s_0 \) outside of \( X \)) is detrimental since \( x \) can actually be reached in 1 step from \( y \). Formally, we can easily prove that \( V^*_X(s_0 \rightarrow x) = V^*_X(s_0 \rightarrow x) = l + 1 \), which grows arbitrarily as \( l \) increases.](attachment:figure3.png)
Moreover, the gap between the two quantities may be arbitrarily large. We restrict the attention to SSP-MDP which enables to track the number of collected samples as $K$ increases. The (possibly unbounded) value function $\pi$ is the set of actions, $p$ is the transition function and $c$ is the cost function. The goal state $s^\dagger$ is zero-cost and absorbing, i.e., $p(s^\dagger|s^\dagger, a) = 1$ and $c(s^\dagger, a) = 0$ for any $a \in A$.

The (possibly unbounded) value function (also called expected cost-to-go) of any policy $\pi \in \Pi$ starting from state $s_0$ is defined as

$$V^\pi(s_0) := \mathbb{E} \left[ \sum_{t=0}^{\infty} c(s_t, \pi(s_t)) \mid s_0 \right] = \mathbb{E} \left[ \tau_{s(s_0 \rightarrow s^\dagger)} \sum_{t=1}^{\infty} c(s_t, \pi(s_t)) \mid s_0 \right].$$

Assumption 2. We restrict the attention to SSP-MDP $M$ (see Def. 7) such that, for any $(s, a) \in S^\dagger \times A$, $c(s, a) \in [c_{\min}, 1]$ with $c_{\min} > 0$. (Note that having positive costs ensures that for any non-proper policy $\pi$ there exists a state $s$ with $V^\pi(s) = +\infty$.) Moreover, we assume that there exists at least one proper policy (i.e., that reaches the goal state $s^\dagger$ with probability one starting from any state in $S^\dagger$).

The procedure $\text{VI}_{\text{SSP}}$ considers the following inputs: a goal $s^\dagger$, non-goal states $S^\dagger$, a known model $p$ and a known cost function $c$, with (non-goal) costs lower bounded by $c_{\min} > 0$. $\text{VI}_{\text{SSP}}$ outputs a vector $u$ (of size $|S^\dagger|$) and a policy $\pi$ which is greedy w.r.t. the vector $u$.

The optimal Bellman operator is defined as follows for any vector $u$ and any non-goal state $s \in S^\dagger$

$$L u(s) := \min_{a \in A} \left\{ c(s, a) + \sum_{s' \in S^\dagger} p(s'|s, a) u(s') \right\}.$$
Algorithm 2: VI\textsubscript{SSP}

\textbf{Input:} Non-goal states \( S^1 \), action set \( A \), transitions \( p \), costs \( c \) and accuracy \( \gamma \)

\textbf{Output:} Value vector \( u \) and greedy policy \( \pi \)

1. Define \( Lu(s) := \min_{a \in A} \left\{ c(s, a) + \sum_{s' \in S^1} p(s'|s, a)u(s') \right\} \)
2. Set \( u_0 = 0_{S^1} \) and \( j = 0 \)
3. \( u_1 = Lu_0 \)
4. while \( \|u_{j+1} - u_j\|_{\infty} > \gamma \) do
   5. \( u_{j+1} = Lu_j \)
6. Set \( u := u_j \) and \( \pi(s) \in \arg\min_{a \in A} \left\{ c(s, a) + \sum_{s' \in S^1} p(s'|s, a)u(s') \right\} \) for any \( s \in S^1 \cup \{s^1\} \)

Note that by definition, \( V^\pi(s^1) = 0 \) for any \( \pi \). We perform a value iteration (VI) scheme over this operator as explained in [e.g., 29, 34, 27]. Namely, we consider initial vector \( u_0 := 0 \) and set iteratively \( u_{i+1} := Lu_i \) (see Alg. 2). For a predefined VI precision \( \gamma > 0 \), the stopping condition is reached for the first iteration \( j \) such that \( \|u_{j+1} - u_j\|_{\infty} \leq \gamma \). The policy is then selected to be the greedy policy w.r.t. the vector \( u := u_j \), i.e.,

\[
\forall s \in S^1 \cup \{s^1\}, \quad \pi(s) \in \arg\min_{a \in A} \left\{ c(s, a) + \sum_{s' \in S^1} p(s'|s, a)u(s') \right\}.
\] (7)

Importantly, while \( u \) is not the value function of \( \pi \), both quantities can be related according to the following lemma.

**Lemma 2.** Consider an SSP-MDP \( M = (S^1, A, s^1, p, c) \) defined as in Def. 7 and satisfying Asm. 2. Let \( (u, \pi) = \text{VI}_{\text{SSP}}(S^1, A, c, \gamma) \) be the solution computed by \( \text{VI}_{\text{SSP}} \). Denote by \( V^\pi \) the true value function of \( \pi \) and by \( V^* = V^\pi = \mathcal{L}V^* \) the optimal value function. The following component-wise inequalities hold

- \( u \leq V^* \leq V^\pi \).
- If the VI precision level verifies \( \gamma \leq \frac{c_{\text{min}}}{2} \), then \( V^\pi \leq \left( 1 + \frac{2\gamma}{c_{\text{min}}} \right) u \).

**Proof.** The result can be obtained by adapting [27, Lem. 4 & App. E]. For the first inequality, given that we consider the initial vector \( u_0 = 0 \), we know that \( 0 \leq V^* \) with \( V^* = \mathcal{L}V^* \) by definition. By monotonicity of the operator \( \mathcal{L} [25, 26] \), we obtain \( u_j \leq V^* \leq V^\pi \). As for the second inequality, we introduce the following Bellman operators of a deterministic policy \( \pi \) for any vector \( u \) and state \( s \),

\[
\mathcal{L}^\pi u(s) := c(s, \pi(s)) + \sum_{s' \in S} p(s'|s, \pi(s))u(s'),
\]

\[
\mathcal{T}^\pi u(s) := c(s, \pi(s)) - \gamma + \sum_{s' \in S} p(s'|s, \pi(s))u(s') > 0.
\]

Note that the SSP problem defined by the operator \( \mathcal{T}^\pi \) satisfies Asm. 2 since i) it has positive costs due to the condition \( \gamma \leq \frac{c_{\text{min}}}{2} \) and ii) the fact that \( M \) satisfies Asm. 2 guarantees the existence of at least one proper policy in the model \( p \). We can write component-wise

\[
\mathcal{T}^\pi u_j = \mathcal{L}^\pi u_j - \gamma \overset{(a)}{=} \mathcal{L}u_j - \gamma \overset{(b)}{=} u_j,
\]

where (a) uses that \( \pi \) is the greedy policy w.r.t. \( u_j \) and (b) stems from the chosen stopping condition which yields \( \mathcal{L}u_j \leq u_j + \gamma \). By monotonicity of the operator \( \mathcal{T}^\pi \), we have for all \( m > 0 \), \( (\mathcal{T}^\pi)^m u_j \leq u_j \). The asymptotic convergence of the operator in an SSP problem satisfying Asm. 2 (see e.g., [26, Prop. 2.2.1]) guarantees that taking the limit \( m \to +\infty \) yields \( W^\pi \leq u_j \), where \( W^\pi \) is defined as the value function of policy \( \pi \) in the model \( p \) with \( \gamma \) subtracted to all the costs, i.e.,

\[
W^\pi(s) := \mathbb{E} \left[ \sum_{t=1}^{\tau^\pi(s)} (c(s_t, \pi(s_t)) - \gamma)|s_1 = s \right] = V^\pi(s) - \gamma \mathbb{E}[\tau^\pi(s)],
\]
where $\tau_\pi(s)$ denotes the (random) hitting time of policy $\pi$ to reach the goal starting from state $s$. Moreover, we have $c_{\min}E[\tau_\pi(s)] \leq V_\pi(s) \leq c_{\max}E[\tau_\pi(s)]$. Putting everything together, we thus get

$$(1 - \frac{c}{c_{\min}})V_\pi \leq u_j.$$ 

Since $\gamma \leq \frac{c_{\min}}{2}$, we ultimately obtain

$$V_\pi \leq \frac{1}{1 - \frac{c}{c_{\min}}} u_j \leq \left(1 + \frac{2\gamma}{c_{\min}}\right) u_j,$$

where the last inequality uses the fact that $\frac{1}{1-x} \leq 1 + 2x$ holds for any $0 \leq x \leq \frac{1}{2}$.  

\[\square\]

**B.2 Computation of Optimistic Model in Unknown SSP**

Consider an SSP problem $M$ defined as in Asm. 2. Consider that, at any given stage of the learning process, the agent is equipped with $N(s, a)$ samples at each state-action pair. A method to compute an optimistic model $\tilde{p}$ is provided in [28], which we recall below.

Denote by $\tilde{p}$ the current empirical average of transitions: $\tilde{p}(s'|s, a) = N(s, a, s')/N(s, a)$, and set $\tilde{\sigma}^2(s'|s, a) := \tilde{p}(s'|s, a)(1 - \tilde{p}(s'|s, a))$ as well as $N^+(s, a) := \max\{1, N(s, a)\}$. For any $(s, a, s') \in S^1 \times A \times S^1$, the empirical Bernstein inequality [35, 36] is leveraged to select the following confidence intervals (with probability at least $1 - \delta$) on the transition probabilities

$$\beta(s, a, s') := 2\sqrt{\frac{\tilde{\sigma}^2(s'|s, a)}{N^+(s, a)}} \log \left(\frac{2SAN^+(s, a)}{\delta}\right) + \frac{6 \log \left(\frac{2SAN^+(s, a)}{\delta}\right)}{N^+(s, a)},$$

and $\beta(s, a, s^1) := \sum_{s' \in S^1} \beta(s, a, s')$. The selection of the optimistic model $\tilde{p}$ is as follows: the probability of reaching the goal $s^1$ is maximized at every state-action pair, which implies minimizing the probability of reaching all other states and setting them at the lowest value of their confidence range. Formally, we set for all $(s, a, s') \in S^1 \times A \times S^1$,

$$\tilde{p}(s'|s, a) := \max \left\{\tilde{p}(s'|s, a) - \beta(s, a, s'), 0\right\},$$

and $\tilde{p}(s^1|s, a) := 1 - \sum_{s' \in S^1} \tilde{p}(s'|s, a)$.

**B.3 Combining the two: Optimistic Value Iteration for SSP (OVI$_{SSP}$)**

OVI$_{SSP}$ first computes an optimistic model $\tilde{p}$ leveraging App. B.2, and it then runs the VI$_{SSP}$ procedure of App. B.1 in the model $\tilde{p}$, i.e., $(\tilde{u}, \tilde{\pi}) = $ VI$_{SSP}(S^1, A, s^1, \tilde{p}, c)$. This outputs an optimistic pair $(\tilde{u}, \tilde{\pi})$ composed of the VI vector $\tilde{u}$ and the policy $\tilde{\pi}$ that is greedy w.r.t. $\tilde{u}$ in the model $\tilde{p}$. The OVI$_{SSP}$ scheme is recapped in Fig. 4.

**C Useful Result: Simulation Lemma for SSP**

Consider a stochastic shortest-path (SSP) instance (see Def. 7) that satisfies Asm. 2. We denote by $A = |A|$ the number of actions, $S = |S|$ the number of non-goal states, $g \notin S$ the (zero-cost and absorbing) goal state, $p$ the unknown transitions and $c$ the known cost function. We assume that $0 < c(s, a) \leq 1$ for all $(s, a) \in S \times A$, and set $c_{\min} := \min_{s, a} c(s, a) > 0$. We also set $S' := S \cup \{g\}$.
Recall that the goal state is zero-cost (i.e., \(c(g, a) = 0\)) and absorbing (i.e., \(p(g|g, a) = 1\)), and that the value function of a policy amounts to the expected cumulative costs following this policy until reaching the goal.

**Definition 8.** For any model \(p\) and \(\eta > 0\), we introduce the set of models close to \(p\) w.r.t. the \(\ell_1\)-norm on the non-goal states as follows

\[
P_{\eta}^{(p)} := \left\{ p' \in \mathbb{R}^{S' \times A \times S'} : \forall (s, a) \in S \times A, \ p'(|s, a) \in \Delta(S'), \ p(g|g, a) = 1, \right. \\
\left. \sum_{y \in S} |p(y|s, a) - p'(y|s, a)| \leq \eta \right\}.
\]

**Lemma 3 (Simulation Lemma for SSP).** Consider any model \(p\) and \(p' \in P_{\eta}^{(p)}\) such that, for each model, there exists at least one proper policy w.r.t. the goal state \(g\). Consider any policy \(\pi\) that is proper in \(p'\), with value function denoted by \(V'_{\pi}\), such that the following condition is verified

\[
\eta \|V'\|_{\infty} \leq 2c_{\min}.
\]

Then \(\pi\) is proper in \(p\) (i.e., its value function verifies \(V_{\pi} < +\infty\) component-wise), and we have

\[
\forall s \neq g, \ V_{\pi}(s) \leq \left(1 + \frac{2\eta \|V'\|_{\infty}}{c_{\min}}\right) V'_{\pi}(s),
\]

and conversely,

\[
\forall s \neq g, \ V'_{\pi}(s) \leq \left(1 + \frac{\eta \|V'\|_{\infty}}{c_{\min}}\right) V_{\pi}(s).
\]

Combining the two inequalities above yields

\[
\|V_{\pi} - V'_{\pi}\|_{\infty} \leq \frac{7\eta \|V'\|_{\infty}^2}{c_{\min}}.
\]

**Proof.** The proof of Lem. 3 requires a result of [37] recalled in Lem. 4 and can be seen as a generalization of [28, Lem. B.4]. First, let us assume that \(\pi\) is proper in the model \(p'\). This implies that its value function, denoted by \(V'\), is bounded component-wise. Moreover, for any non-goal state \(s \in S\), the Bellman equation holds as follows

\[
V'(s) = c(s, \pi(s)) + \sum_{y \in S} p'(y|s, \pi(s))V'(y)
\]

\[
= c(s, \pi(s)) + \sum_{y \in S} p(y|s, \pi(s))V'(y) + \sum_{y \in S} (p'(y|s, \pi(s)) - p(y|s, \pi(s)))V'(y).
\]

By successively using Hölder’s inequality and the facts that \(p' \in P_{\eta}^{(p)}\) and \(c(s, \pi(s)) \geq c_{\min}\), we get

\[
V'(s) \geq c(s, \pi(s)) - \eta \|V'\|_{\infty} + p(|s, \pi(s))^{\top} V' \geq c(s, \pi(s)) \left(1 - \frac{\eta \|V'\|_{\infty}}{c_{\min}}\right) + p(|s, \pi(s))^{\top} V'.
\]

Let us now introduce the vector \(V'' := \left(1 - \frac{\eta \|V'\|_{\infty}}{c_{\min}}\right)^{-1}V'\). Then for all \(s \in S\),

\[
V''(s) \geq c(s, \pi(s)) + p(|s, \pi(s))^{\top} V''.
\]

Hence, from Lem. 4, \(\pi\) is proper in \(p\) (i.e., \(V < +\infty\)), and we have

\[
V \leq V'' \leq \left(1 + 2 \frac{\eta \|V'\|_{\infty}}{c_{\min}}\right) V',
\]

where the last inequality stems from condition (8) and the fact that \(\frac{1}{1-x} \leq 1 + 2x\) holds for any \(0 \leq x \leq \frac{1}{2}\). Conversely, analyzing Eq. 9 from the other side, we get

\[
V'(s) \leq c(s, \pi(s)) \left(1 + \frac{\eta \|V'\|_{\infty}}{c_{\min}}\right) + p(|s, \pi(s))^{\top} V'.
\]
Let us now introduce the vector $V'' := \left(1 + \frac{\eta\|V'\|_{\infty}}{c_{\min}}\right)^{-1} V'$. Then

$$V''(s) \leq c(s, \pi(s)) + p(\cdot|s, \pi(s))^\top V''.$$  

We then obtain in the same vein as Lem. 4 (by leveraging the monotonicity of the Bellman operator \(E^\pi U(s) := c(s, \pi(s)) + p(\cdot|s, \pi(s))^\top U\)) that $V'' \leq V'$, and therefore

$$V' \geq \left(1 + \frac{\eta\|V'\|_{\infty}}{c_{\min}}\right)^{-1} V.$$  

Combining Eq. 10 and 11 yields component-wise

$$\|V - V'\|_{\infty} \leq 2\eta\|V'\|_{\infty}\|V''\|_{\infty} + \eta\|V'\|_{\infty}\|V\|_{\infty} \leq 7\eta\|V'\|_{\infty}^2,$$

where the last inequality uses that $\|V\|_{\infty} \leq 5\|V'\|_{\infty}$ which stems from plugging condition (8) into Eq. 10.

Note that here \(p\) and \(p'\) play symmetric roles; we can perform the same reasoning in the case where \(\pi\) is proper in the model \(p\) and it would yield an equivalent result by switching the dependencies on \(V\) and \(V''\).

**Lemma 4 ([37], Lem. 1).** In an SSP-MDP satisfying Asm. 2, let \(\pi\) be any policy, then

- If there exists a vector \(U : S \rightarrow \mathbb{R}\) such that \(U(s) \geq c(s, \pi(s)) + \sum_{s' \in S} \pi(s'|s, \pi(s))U(s')\) for all \(s \in S\), then \(\pi\) is proper, and \(V^\pi\) the value function of \(\pi\) is upper bounded by \(U\) component-wise, i.e., \(V^\pi(s) \leq U(s)\) for all \(s \in S\).
- If \(\pi\) is proper, then its value function \(V^\pi\) is the unique solution to the Bellman equations \(V^\pi(s) = c(s, \pi(s)) + \sum_{s' \in S} \pi(s'|s, \pi(s))V^\pi(s')\) for all \(s \in S\).

## D Proof of Theorem 1 (Sample Complexity Analysis of DisCo)

### D.1 Computation of the Optimistic Policies

At each round \(k\), for each goal state \(s^g \in W_k\), DisCo computes an optimistic goal-oriented policy associated to the MDP \(M^*_k(s^g)\) constructed as in Def. 6. This MDP is defined over the entire state space \(S\) and restricts the action to the only action RESET outside \(K_k\). We can build an equivalent MDP by restricting the focus on \(K_k\). To this end, we define the following SSP-MDP.

**Definition 9.** Define \(M^*_k(s^g) := (S^*_k, A^*_k, c^*_k, p^*_k)\) where \(S^*_k := K_k \cup \{s^g, \pi\} \times \{\text{RESET}\}\) and \(S^*_k = |S^*_k| = |K_k| + 2\). State \(x\) is a meta-state that encapsulates all the states that have been observed so far and are not in \(K_k\). The action space \(A^*_k\) is such that \(A^*_k(s) = A\) for all states \(s \in K_k\) and \(A^*_k(s) = \{\text{RESET}\}\) for \(s \in \{s^g, \pi\}\). The cost function is \(c^*_k(x, a) = 0\) for any \(a \in A^*_k(x)\) and \(c^*_k(s, a) = 1\) everywhere else. The transition function is defined as \(p^*_k(s'|s, a) = p^*_k(s_0|x, a) = 1\) for any \(a\), \(p^*_k(y|x, a) = p(y|x, a)\) for any \((s, a, y) \in K_k \times A \times (K_k \cup \{s^g\})\) and \(p^*_k(x|s, a) = 1 - \sum_{y \in K_k \cup \{s^g\}} p^*_k(y|x, a)\).

Note that solving \(M^*_k\) yields a policy effectively restricted to the set \(K_k\) insofar as we can interpret the meta-state \(x\) as \(S \setminus \{K_k \cup \{s^g\}\}\). Since \(p\) is restricted, we cannot construct \(M^*_k(s^g)\). Let \(N_k\) be the state-action counts accumulated up until now. We denote by \(\tilde{p}^*_k\) the “global” empirical estimates, i.e., \(\tilde{p}^*_k(y|x, a) = N_k(s, a, y)/N_k(s, a)\). Given them, we define the “restricted” empirical estimates \(\hat{p}^*_k\) as follows: \(\hat{p}^*_k(y|x, a) := \tilde{p}^*_k(y|x, a)\) for any \((s, a, y) \in K_k \times A \times (K_k \cup \{s^g\})\) and \(\hat{p}^*_k(x|s, a) := 1 - \sum_{y \in K_k \cup \{s^g\}} \hat{p}^*_k(y|x, a)\). Denoting \(N^+_k(s, a) := \max\{1, N_k(s, a)\}\), we then define the following bonuses for any \((s, a, y) \in K_k \times A \times (K_k \cup \{s^g\})\),

\[
\beta_k(s, a, y) := \sqrt{\frac{\tilde{p}_k(y|x, a)(1 - \tilde{p}_k(y|x, a))}{N^+_k(s, a)}} \log \left(\frac{2SAN^+_k(s, a)}{\delta}\right) + \frac{6\log\left(\frac{2SAN^+_k(s, a)}{\delta}\right)}{N^+_k(s, a)},
\]

\[
\beta_k(s, a) := \sum_{y \in K_k \cup \{s^g\}} \beta_k(s, a, y).
\]
Moreover, we set the uncertainty about the MDP at the meta-state $x$ and at the goal state $s^\dagger$ to 0 by construction (since their outgoing transitions are deterministic, respectively to $s_0$ and $s^\dagger$).

We now leverage the optimistic construction mentioned in App. B.1.

**Definition 10.** We denote by $\tilde{M}_k^\dagger(s^\dagger) = (S_k^\dagger, A_k^\dagger, c_k^\dagger, \tilde{p}_k^\dagger)$ the optimistic MDP associated to $M_k^\dagger(s^\dagger)$ defined in Def. 9. Then, $\forall (s, a) \in K_k \times A$,

\[
\tilde{p}_k^\dagger(y|s, a) := \max\{\tilde{p}_k(y|s, a) - \beta_k(s, a, y), 0\}, \quad \forall y \in K_k \cup \{x\},
\]

\[
\tilde{p}_k^\dagger(s^\dagger|s, a) := 1 - \sum_{y \in K_k \cup \{x\}} \tilde{p}_k^\dagger(y|s, a),
\]

\[
\tilde{p}_k^\dagger(s^\dagger|s^\dagger, a) = \tilde{p}_k^\dagger(s_0|x, a) = 1.
\]

Given this MDP, we can compute the optimistic value vector $\hat{u}_k^\dagger$ and policy $\hat{\pi}_k^\dagger$ using value iteration for SSP: $(\hat{u}_k^\dagger, \hat{\pi}_k^\dagger) = \text{VI}_{\text{SSP}}(S_k^\dagger, A_k^\dagger, s_0^\dagger, \tilde{p}_k^\dagger, \tilde{\pi})$. We summarize the construction of the optimistic model and the computation of value function and policy in Alg. 3 (OVI$_{\text{SSP}}$).

**Remark.** Note that the structure of the problem does not appear to allow for variance-aware improvements in the analysis of Thm. 1 (specifically, when the analysis will apply an SSP simulation lemma argument). Indeed, given the possibly large number of states in the total environment $S$, the computation of the optimistic policies requires the construction of the meta-state $x$ that encapsulates all the states in $S \setminus \{K_k \cup \{s^\dagger\}\}$, where $s^\dagger$ is the candidate goal state considered at round $k$. As a result, the uncertainty on the transitions reaching $x$ needs to be summed over multiple states, as shown in Eq. 13. This extra uncertainty at a single state in the induced MDP has the effect of canceling out Bernstein techniques seeking to lower the prescribed requirement of the state-action samples that the algorithm should collect. In turn this implies that such variance-aware techniques would not lead to any improvement in the final sample complexity bound.

### D.2 High-Probability Event

**Lemma 5.** It holds with probability at least $1 - \delta$ that for any time step $t \geq 1$ and for any state-action pair $(s, a)$ and next state $s^\prime$,

\[
|\tilde{p}_t(s'|s, a) - p(s'|s, a)| \leq 2 \sqrt{\frac{\tilde{\sigma}_t^2(s'|s, a) + \sigma_t^2(s'|s, a)}{N_t^+(s, a)} + \frac{6 \log \left(\frac{2SAN_t(s, a)}{\delta} \right)}{N_t^+(s, a)}}\frac{2SAN_t(s, a)}{\delta},
\]

where $N_t^+(s, a) := \max\{1, N_t(s, a)\}$ and where $\tilde{\sigma}_t^2$ are the population variance of transitions, i.e., $\tilde{\sigma}_t^2(s'|s, a) := \tilde{p}_t(s'|s, a)(1 - \tilde{p}_t(s'|s, a))$.

**Proof.** The confidence intervals in Eq. 17 are constructed using the empirical Bernstein inequality, which guarantees that the considered event holds with probability at least $1 - \delta$, see e.g., [38].

Define the set of plausible transition probabilities as

\[
C_k^\dagger := \bigcap_{(s, a) \in S_k^\dagger \times A} C_k^\dagger(s, a),
\]
We fix any round $k$.

We recall notation. Let us fix any round $k$ where

$$C_k(s, a) := \{ \bar{p} \in C \mid \bar{p}(\cdot | s^t, a) = 1_{s^t}, \bar{p}(\cdot | x, a) = 1_{s_0}, |\bar{p}(s' | s, a) - \hat{p}_k(s' | s, a)| \leq \beta_k(s, a, s') \},$$

with $C$ the $S_k$-dimensional simplex and $\hat{p}_k$ the empirical average of transitions.

**Lemma 6.** Introduce the event $\Theta := \bigcap_{k=1}^{\infty} \bigcap_{s \in W_k} \{ p^k_1 \in C_k^1 \}$. Then $\mathbb{P}(\Theta) \geq 1 - \frac{\delta}{3}$.

**Proof.** We have with probability at least $1 - \frac{\delta}{3}$ that, for any $y \neq x$, $|p^k_1(y | s, a) - \hat{p}_k^1(y | s, a)| \leq \beta_k(s, a, y)$ from the empirical Bernstein inequality (see Eq. 17), and moreover $|\hat{p}_k^1(x | s, a) - p^k_1(x | s, a)| = \left| 1 - \sum_{y \in K_k \cup \{ s^t \}} p^k_1(y | s, a) - \left( 1 - \sum_{y \in K_k \cup \{ s^t \}} \hat{p}_k^1(y | s, a) \right) \right| \leq \sum_{y \in K_k \cup \{ s^t \}} |p^k_1(y | s, a) - \hat{p}_k^1(y | s, a)| \leq \beta_k(s, a, x).$ \hfill $\Box$

**Lemma 7.** Under the event $\Theta$, for any round $k$ and any goal state $s^t \in W_k$, the optimistic model $\hat{p}^k_1$ constructed in Def. 10 verifies $\hat{p}^k_1 \in \mathcal{P}_{\eta_k}^{p_1}$, with $\eta_k := 4\beta_k(s, a, x)$ where $\beta_k$ is defined in Eq. 13.

**Proof.** Combining the construction in Def. 10, the proof of Lem. 6 and the triangle inequality yields

$$\sum_{y \in K_k \cup \{ x \}} |\hat{p}^k_1(y | s, a) - p^k_1(y | s, a)| \leq \sum_{y \in K_k \cup \{ x \}} |\hat{p}^k_1(y | s, a) - \hat{p}^k_1(y | s, a)| + |\hat{p}^k_1(y | s, a) - p^k_1(y | s, a)|$$

$$\leq \sum_{y \in K_k \cup \{ x \}} \beta_k(s, a, y) + 2\beta_k(s, a, x)$$

$$\leq 4\beta_k(s, a, x).$$ \hfill $\Box$

Throughout the remainder of the proof, we assume that the event $\Theta$ holds.

### D.3 Properties of the Optimistic Policies and Value Vectors

We recall notation. Let us fix any round $k$ and any goal state $s^t \in W_k$. We denote by $\tilde{\pi}^1_k$ the greedy policy w.r.t. $\tilde{u}^1_k(\cdot \rightarrow s^t)$ in the optimistic model $\hat{p}^k_1$. Let $\tilde{v}^1_k(s \rightarrow s^t)$ be the value function of policy $\tilde{\pi}^1_k$ starting from state $s$ in the model $\hat{p}^k_1$. We can apply Lem. 2 given that the conditions of Asm. 2 hold (indeed, we have $c_{\min} = 1 > 0$ and there exists at least one proper policy to reach the goal state $s^t$ since it belongs to $W_k$). Moreover, we have that $V^*_k(s_0 \rightarrow s^t) \leq V^*_k(s_0 \rightarrow s^t)$ given the way the optimistic model $\hat{p}^k_1$ is computed (i.e., by maximizing the probability of transitioning to the goal at any state-action pair), see [28, Lem. B.12]. Hence we get the two following important properties.

**Lemma 8.** For any round $k$, goal state $s^t \in W_k$ and state $s \in K_k \cup \{ x \}$, we have under the event $\Theta$,

$$\tilde{u}^1_k(s \rightarrow s^t) \leq V^*_k(s \rightarrow s^t).$$

**Lemma 9.** For any round $k$, goal state $s^t \in W_k$ and state $s \in K_k \cup \{ x \}$, we have

$$\tilde{v}^1_k(s \rightarrow s^t) \leq (1 + 2\gamma)\tilde{u}^1_k(s \rightarrow s^t).$$

### D.4 State Transfer from $U$ to $K$ (step $\oplus$)

We fix any round $k$ and any goal state $s^t \in W_k$ that is added to the set of “controllable” states $K$, i.e., for which $\tilde{u}^1_k(s_0 \rightarrow s^t) \leq L$.

**Lemma 10.** Under the event $\Theta$, we have both following inequalities

$$\begin{align*}
\tilde{v}^1_k(s_0 \rightarrow s^t) &\leq L + \epsilon, \\
\tilde{v}^1_k(s_0 \rightarrow s^t) &\leq V^*_k(s_0 \rightarrow s^t) + \epsilon.
\end{align*}$$

In particular, the first inequality entails that $s^t \in S^*_L$, which justifies the validity of the state transfer from $U$ to $K$. 

20
Proof. We have

\[ \bar{v}_k^1(s_0 \to s^1) \leq (1 + 2\gamma)\bar{u}_k^1(s_0 \to s^1) \leq \begin{cases} \leq L + \frac{\varepsilon}{3} \\ \leq V_{k_{\star}}^*(s_0 \to s^1) + \frac{\varepsilon}{3} \end{cases}, \]  

(18)

where inequality (a) comes from Lem. 9, inequality (b) combines the algorithmic condition \( \bar{u}_k^1(s_0 \to s^1) \leq L \) and the VI precision level \( \gamma := \frac{3\varepsilon}{L} \), and finally inequality (c) combines Lem. 8 and the VI precision level. Moreover, for any state in \( K_k \),

\[ \bar{v}_k^1(s \to s^1) \leq V_{k_{\star}}^*(s \to s^1) + 1 + \frac{\varepsilon}{3} \leq \bar{v}_k^1(s_0 \to s^1) + 1 + \frac{\varepsilon}{3}, \]

where (a) comes from Lem. 8 and (b) stems from the presence of the RESET action (Asm. 1).

We now provide the exact choice of allocation function \( \phi \) in Alg. 1. We introduce

\[ \gamma := \frac{2\varepsilon}{12(L + 1 + \varepsilon)(L + \frac{\varepsilon}{3})}. \]

(Note that \( \gamma = O(\varepsilon/L^2) \).) We set the following requirement of samples for each state-action pair \( (s, a) \) at round \( k \),

\[ n_k = \phi(K_k) = \left[ \frac{57X_k^2}{\gamma^2} \left( \frac{8eX_k\sqrt{2SA}}{\sqrt{\delta\gamma}} \right) \right]^2 + \frac{24|S^1_k|}{\gamma} \log \left( \frac{24|S^1_k||SA|}{\delta\gamma} \right), \]  

(19)

where we define

\[ X_k := \max_{(s, a) \in S_k^1 \times \mathcal{A}} \sum_{s' \in S_k^1} \sqrt{\hat{\sigma}_k^2(s'|s, a)}, \]

with \( \hat{\sigma}_k^2(s'|s, a) := \hat{\pi}_k^1(s'|s, a)(1 - \hat{\pi}_k^1(s'|s, a)) \) the estimated variance of the transition from \( (s, a) \) to \( s' \). Leveraging the empirical Bernstein inequality (Lem. 5) and performing simple algebraic manipulations (see e.g., [39, Lem. 8 and 9]) yields that \( \beta_k(s, a, x) \leq \gamma \). From Lem. 7, this implies that \( \hat{\pi}_k^1 \in \mathcal{P}^{(\eta)}_\gamma \) with \( \eta := 4\gamma \). We can then apply Lem. 3 (whose condition 8 is verified), which gives

\[ v_k^1(s_0 \to s^1) \leq (1 + \eta\|\bar{v}_k^1(\cdot \to s^1)\|_\infty)\bar{v}_k^1(s_0 \to s^1) \leq (1 + \eta(L + 1 + \varepsilon))\bar{v}_k^1(s_0 \to s^1) \leq \bar{v}_k^1(s_0 \to s^1) + \frac{2\varepsilon}{3}, \]

where the last inequality uses that \( \eta(L + 1 + \varepsilon)(L + \frac{\varepsilon}{3}) = \frac{2\varepsilon}{3} \) by definition of \( \gamma \). Plugging in Eq. 18 yields the sought-after inequalities.

\[ \square \]

D.5 Termination of the Algorithm

Lemma 11 (Variant of Lem. 17 of [1]). Suppose that for every state \( s \in S \), each action \( a \in \mathcal{A} \) is executed \( b \geq \lceil L\log(3LA|S|) \rceil \) times. Let \( \mathcal{S}'_{s, a} \) be the set of all next states visited during the \( b \) executions of \( (s, a) \). Denote by \( \Lambda \) the complementary of the event

\[ \left\{ 3(s', s, a) \in S^2 \times \mathcal{A} : p(s'|s, a) \geq \frac{1}{L} \land s' \notin S'_{s, a} \right\}. \]

Then \( \mathbb{P}(\Lambda) \geq 1 - \frac{\varepsilon}{4} \).

Lemma 12. Under the event \( \Theta \cap \Lambda \), for any round \( k \), either \( S_{\Lambda}^r \subseteq K_k \), or there exists a state \( s^1 \in S_{\Lambda}^r \setminus K_k \) such that \( s^1 \in \mathcal{W}_k \) and is \( L \)-controllable with a policy restricted to \( K_k \). Moreover, \( |\mathcal{W}_k| \leq 2LA|K_k| \).
Proof of Lem. 12. Consider a round \( k \) such that \( S_L^+ \setminus K_k \) is non-empty. Due to the incremental construction of the set \( S_L^+ \) (Def. 4), there exists a state \( s^1 \in S_L^- \) and a policy restricted to \( K_k \) that can reach \( s^1 \) in at most \( L \) steps (in expectation). Hence there exists a state-action pair \((s, a) \in K_k \times \mathcal{A}\) such that \( p(s^1|s, a) \geq \frac{1}{L} \). Since \( \phi(K_k) \geq \left[ L \log \left( \frac{3L\mathcal{A}\mathcal{S}}{\varepsilon} \right) \right] \) samples are available at each state-action pair, according to Lem. 11, we get that, under the event \( \Lambda \), \( s^1 \) is found during the sample collection procedure for the state-action pair \((s, a) \) (step 3), which implies that \( s^1 \in \mathcal{U}_k \).

Moreover, the choice of allocation function \( \phi \) guarantees in particular that there are more than \( \Omega(\frac{4L^2}{\varepsilon^2} \log(\frac{2L\mathcal{A}\mathcal{S}}{\varepsilon})) \) samples available at each state-action pair \((s, a) \in K_k \times \mathcal{A}\). From the empirical Bernstein inequality of Eq. 17, we thus have that \( |p(s^1|s, a) - \hat{p}_k(s^1|s, a)| \leq \frac{\varepsilon}{2L} \) under the event \( \Theta \). Consequently we have

\[
\hat{p}_k(s^1|s, a) \geq \frac{1}{L} - |p(s^1|s, a) - \hat{p}_k(s^1|s, a)| \geq \frac{1 - \frac{\varepsilon}{2L}}{L},
\]

which implies that \( s^1 \in \mathcal{W}_k \). Furthermore, we can decompose \( \mathcal{W}_k \) the following way

\[
\mathcal{W}_k = \bigcup_{(s, a) \in K_k \times \mathcal{A}} \mathcal{Y}_k(s, a),
\]

where we introduce the subset

\[
\mathcal{Y}_k(s, a) := \left\{ s' \in \mathcal{U}_k : \hat{p}_k(s'|s, a) \geq \frac{1 - \frac{\varepsilon}{2L}}{L} \right\}.
\]

We then have

\[
1 = \sum_{s' \in \mathcal{S}} \hat{p}_k(s'|s, a) \geq \sum_{s' \in \mathcal{Y}_k(s, a)} \hat{p}_k(s'|s, a) \geq \frac{1 - \frac{\varepsilon}{2L}}{L} |\mathcal{Y}_k(s, a)|.
\]

We conclude the proof by writing that

\[
|\mathcal{W}_k| \leq \sum_{(s, a) \in K_k \times \mathcal{A}} |\mathcal{Y}_k(s, a)| \leq \frac{L}{1 - \frac{\varepsilon}{2L}} A|K_k| \leq 2LA|K_k|,
\]

where the last inequality uses that \( \varepsilon \leq 1 \) (from line 2 of Alg. 1). \( \square \)

Lemma 13. Under the event \( \Theta \cap \Lambda \), when either condition STOP1 or STOP2 is triggered (at a round indexed by \( K \)), we have \( S_L^+ \subseteq K_K \).

Proof. If condition STOP1 is triggered, Lem. 12 immediately guarantees that \( S_L^+ \subseteq K_K \) under the event \( \Lambda \). If condition STOP2 is triggered, we have for all \( s \in \mathcal{W}_K \), \( \hat{u}_s(s_0 \rightarrow s) > L \). From Lem. 8 this means that, under the event \( \Theta \), for all \( s \in \mathcal{W}_K \), \( V^*_{K_K}(s_0 \rightarrow s) > L \). Hence none of the states in \( \mathcal{W}_K \) can be reached in at most \( L \) steps (in expectation) with a policy restricted to \( K_K \). We conclude the proof using Lem. 12. \( \square \)

Lemma 14. Under the event \( \Theta \cap \Lambda \), when DiSCO terminates at round \( K \), for any state \( s \in K_K \), the policy \( \pi_s \) computed during step \( \S \) verifies

\[
v_{\pi_s}(s_0 \rightarrow s) \leq \min_{\pi \in \Pi(S_L^+)} v_{\pi}(s_0 \rightarrow s) + \varepsilon.
\]

Moreover, we have that \( S_L^+ \subseteq K_K \subseteq S_{L+\varepsilon}^+ \).

Proof. Assume that the event \( \Theta \cap \Lambda \) holds. Then when the final set \( K_K \) is considered and the new policies are computed using all the samples, Lem. 10 yields for all \( s \in K_K \),

\[
v_{\pi_s}(s_0 \rightarrow s) \leq \min_{\pi \in \Pi(K_K)} v_{\pi}(s_0 \rightarrow s) + \varepsilon.
\]

Moreover Lem. 13 entails that \( K_K \supseteq S_L^+ \). This implies from Lem. 1 that

\[
\min_{\pi \in \Pi(K_K)} v_{\pi}(s_0 \rightarrow s) \leq \min_{\pi \in \Pi(S_L^+)} v_{\pi}(s_0 \rightarrow s),
\]

which means that \( K_K \subseteq S_{L+\varepsilon}^+ \). \( \square \)
D.6 High Probability Bound on the Sample Collection Phase (step 3)

Denote by \( K \) the (random) index of the last round during which the algorithm terminates. We focus on the sample collection procedure for any state \( s \in K^{-1} \). We denote by \( k_s \) the index of the round during which \( s \) was added to the set of “controllable” states \( K \). To collect samples at state \( s \), the learner uses the shortest-path policy \( \pi_s \). We say that an attempt to collect a specific sample is a rollout. We denote by \( Z_K := |K| A N_K \) the total number of samples that the learner needs to collect. As such, at most \( Z_K \) rollouts must take place. Assume that the event \( \Theta \) holds. Then from Lem. 14, we have \( K_K \subseteq S_{L+\varepsilon} \). Hence, denoting \( S_{L+\varepsilon} := |S_{L+\varepsilon}| \), we have \( Z_K \leq Z_{L+\varepsilon} := S_{L+\varepsilon} A \Phi(S_{L+\varepsilon}) \).

The following lemma provides a high-probability upper bound on the time steps required to meet the sampling requirements.

**Lemma 15.** Assume that the event \( \Theta \) holds. Set

\[
\psi := 4(L + \varepsilon + 1) \log \left( \frac{6Z_{L+\varepsilon}}{\delta} \right),
\]

and introduce the following event

\[
\mathcal{T} := \left\{ \exists \text{ one rollout (with goal state } s \text{) s.t. } \tau_{\pi_s}(s_0 \rightarrow s) > \psi \right\}.
\]

We have \( P(\mathcal{T}) \leq \frac{\delta}{3} \).

**Proof.** Assume that the event \( \Theta \) holds. Leveraging a union bound argument and applying Lem. 16 to policy \( \pi_s \) which verifies \( v_{\pi_s}(s' \rightarrow s) \leq L + \varepsilon + 1 \) for any \( s' \in K_K \), we get

\[
P(\mathcal{T}) \leq \sum_{\text{rollouts}} 2 \exp \left( -\frac{\psi}{4(L + \varepsilon + 1)} \right) \leq 2Z_{L+\varepsilon} \exp \left( -\frac{\psi}{4(L + \varepsilon + 1)} \right) \leq \frac{\delta}{3},
\]

where the last inequality comes from the choice of \( \psi \).

**Lemma 16** ([28], Lem. B.5). Let \( \pi \) be a proper policy such that for some \( d > 0 \), \( V_\pi(s) \leq d \) for every non-goal state \( s \). Then the probability that the cumulative cost of \( \pi \) to reach the goal state from any state \( s \) is more than \( m \), is at most \( 2e^{-m/(4d)} \) for all \( m \geq 0 \). Note that a cost of at most \( m \) implies that the number of steps is at most \( m/d \).

D.7 Putting Everything Together: Sample Complexity Bound

The sample complexity of the algorithm is solely induced by the sample collection procedure (step 3). Recall that we denote by \( K \) the index of the round at which the algorithm terminates. With probability at least \( 1 - \frac{2\delta}{3} \), Lem. 13 holds, and so does the event \( \Theta \). Hence the algorithm discovers a set of states \( K_K \subseteq S_{L}^{\pi_\ast} \). Moreover, from Lem. 14, the algorithm outputs for each \( s \in K_K \) a policy \( \pi_s \) with \( \mathbb{E}[\tau_{\pi_s}(s_0 \rightarrow s)] \leq V_{\pi_s}(s) + \varepsilon \). Hence we also have \( |K_K| \leq S_{L+\varepsilon} := |S_{L+\varepsilon}| \).

We denote by \( Z_K := |K_K| A \phi(K_K) \) the total number of samples that the learner needs to collect. From Lem. 15, with probability at least \( 1 - \frac{2\delta}{3} \), the total sample complexity of the algorithm is at most \( \psi Z_K \), where \( \psi := 4(L + \varepsilon + 1) \log \left( \frac{6Z_{L+\varepsilon}}{\delta} \right) \).

Now, from Eq. 19 there exists an absolute constant \( \alpha > 0 \) such that DisCo selects as allocation function

\[
\phi : X \rightarrow \alpha \cdot \left( \frac{L^2 \hat{\Theta}(X)}{\varepsilon^2} \log^2 \left( \frac{LSA}{\varepsilon \delta} \right) + \frac{L^2 |X|}{\varepsilon} \log \left( \frac{LSA}{\varepsilon \delta} \right) \right),
\]

where

\[
\hat{\Theta}(X) := \max_{(s,a) \in X \times A} \left( \sum_{s' \in X} \sqrt{p(s'|s,a)(1 - \hat{p}(s'|s,a))} \right)^2.
\]

The total requirement is \( \phi(K_K) \). Note that from Cauchy-Schwarz’s inequality, we have

\[
\hat{\Theta}(K_K) \leq \Gamma_K := \max_{(s,a) \in K_K \times A} \| \{p(s'|s,a)\}_{s' \in K_K} \|_0 \leq |K_K|.
\]
Combining everything yields with probability at least \( 1 - \delta \),

\[
\psi Z_K = \tilde{O}
\left(
\frac{L^5 \Gamma K |\mathcal{K}| A}{\varepsilon^2} + \frac{L^3 |\mathcal{K}|^2 A}{\varepsilon}
\right).
\]

We finally use that \( \mathcal{K} \subseteq S_{L + \varepsilon}^\top \) from Lem. 14, which implies that

\[
C_{AX^*}(\text{DisCo}, L, \varepsilon, \delta) = O
\left(
\frac{L^5 \Gamma L_{L + \varepsilon} S_{L + \varepsilon} A}{\varepsilon^2} + \frac{L^3 S_{L + \varepsilon}^2 A}{\varepsilon}
\right),
\]

where \( \Gamma_{L + \varepsilon} := \max_{(s, a) \in S_{L + \varepsilon}^\top \times A} \{ p(s' | s, a) \} s' \in S_{L + \varepsilon}^\top \| 0 \). This concludes the proof of Thm. 1.

D.8 Proof of Corollary 1

The result given in Cor. 1 comes from retracing the analysis of Lem. 14 and therefore Lem. 10 by considering non-uniform costs between \([c_{\min}, 1] \) instead of costs all equal to 1. Specifically, Eq. 20 needs to account for the inverse dependency on \( c_{\min} \) of the simulation lemma of Lem. 3. This induces the final \( \varepsilon / c_{\min} \) accuracy level achieved by the policies output by DisCo. There remains to guarantee that condition 8 of Lem. 3 is verified. In particular the condition holds if \( \eta(L + 1 + \varepsilon) \leq 2c_{\min} \), where \( \eta \) is the model accuracy prescribed in the proof of Lem. 10. We see that this is the case whenever we have \( \varepsilon = O(Lc_{\min}) \) due to the fact that \( \eta = \Omega(\varepsilon / L^2) \).

D.9 Computational Complexity of DisCo

The overall computational complexity of DisCo can be expressed as \( \sum_{k=1}^K |W_k| \cdot C(\text{OVI}_{\text{SSP}}) \), where \( C(\text{OVI}_{\text{SSP}}) \) denotes the complexity of an OVI_{SSP} procedure and where we recall that \( K \) denotes the (random) index of the last round during which the algorithm terminates. Note that it holds with high probability that \( K \leq |S_{L + \varepsilon}^\top| \) and \( |W_k| \leq 2L \mathcal{A} |\mathcal{K}| \leq 2LA|S_{L + \varepsilon}^\top| \). Moreover \( C(\text{OVI}_{\text{SSP}}) \) captures the complexity of the value iteration (VI) algorithm for SSP, which was proved in [34] to converge in time quadratic w.r.t. the size of the considered state space (here, \( \mathcal{K}_k \)) and \( \| V^* \|_\infty / c_{\min} \). Here we have \( c_{\min} = 1 \), and we can easily prove that in all the SSP instances considered by DisCo, the optimal value function \( V^* \) verifies \( \| V^* \|_\infty = O(L^2) \), due to the restriction of the goal state in \( W_k \) (indeed this restriction implies that there exists a state-action pair in \( \mathcal{K}_k \times A \) that transitions to the goal state with probability \( \Omega(1/L) \) in the true MDP). Putting everything together gives DisCo’s computational complexity. Interestingly, we notice that while it depends polynomially on \( S_{L + \varepsilon}, L \) and \( A \), it is independent from \( S \) the size of the global state space.

E The UcbExplore Algorithm [1]

E.1 Outline of the Algorithm

The UcbExplore algorithm was introduced by Lim and Auer [1] to specifically tackle condition AX_1. The algorithm maintains a set \( \mathcal{K} \) of “controllable” states and a set \( \mathcal{U} \) of “ uncontrollable” states. It alternates between two phases of state discovery and policy evaluation. In a state discovery phase, new candidate states are discovered as potential members of the set of controllable states. Any policy evaluation phase is called a round and it relies on an optimistic principle: it attempts to reach an “optimistic” state \( s \) (i.e., the easiest state to reach based on information collected so far) among all the candidate states by executing an optimistic policy \( \pi_s \) that minimizes the optimistic expected hitting time truncated at a horizon of \( \lambda_{\text{Ucb}} := [L + L^2 \varepsilon^{-1}] \). Within the round of evaluation of policy \( \pi_s \), the algorithm proceeds through at most \( \lambda_{\text{Ucb}} := [6L^3 \varepsilon^{-3} \log \{ 16 |\mathcal{K}|^2 \delta^{-1} \}] \) episodes, each of which begins at \( s_0 \) and ends either when \( \pi_s \) successfully reaches \( s \) or when \( \lambda_{\text{Ucb}} \) steps have been executed. If the empirical performance of \( \pi_s \) is poor (measured through a performance check done after each episode), the round is said to have failed. Otherwise, the round is successful which means that \( s \) is controllable and an acceptable policy \( (\pi_s) \) has been discovered. A failure round leads to selecting another candidate state-policy pair for evaluation, while a success round leads to a state discovery phase which in turn adds more candidate states for the subsequent rounds. As explained in App. A, UcbExplore is unable to tackle the more challenging objective AX^*.
We denote by $\Pi_s$ where $e\lambda_{UCB}$ is the actual number of episodes executed in round $k$ and where the reward $r_i \in \{0,1\}$ is equal to 1 only if the state is the goal state. However, upper bounding the regret contribution of failure rounds implies applying a concentration inequality on only specific rounds that are chosen given their empirical performance. Hence Lim and Auer [1, Lem. 18] improperly use a martingale argument to bound a sum whose summands are chosen in a non-martingale way, i.e., depending on their realization.

To avoid the aforementioned issue, one must upper and lower bound the cumulative regret of the entire set of rounds and not only the failure rounds in order to obtain a bound on the number of failure rounds. However, this would yield a sample complexity that has a second term scaling as $O(\varepsilon^{-4})$. Following personal communication with the authors, the fix is to change the definition of regret of a round, making it equal to

$$\sum_{j=1}^{e_k} H_{UCB} - L - \sum_{i=0}^{H_{UCB}-1} r_i,$$

where $s$ is the considered goal state and $\bar{u}_{H_{UCB}}(s_0 \rightarrow s)$ is the optimistic $H_{UCB}$-step reward (where the reward is equal to 1 only at state $s$). With this new definition, it is possible to recover the sample complexity provided in [1] scaling as $O(\varepsilon^{-3})$.

### E.3 Issue with a Possibly Infinite State Space

Lim and Auer [1] claim that their setting can cope with a countable, possibly infinite state space. However, this leads to a technical issue, which has been acknowledged by the authors via personal communication and as of now has not been resolved. Indeed, it occurs when a union bound over the unknown set $U$ is taken to guarantee high-probability statements (e.g., the Lem. 14 or 17 of [1]). Yet for each realization of the algorithm, we do not know what the set $U$, or equivalently $\mathcal{K}$, looks like, hence it is improper to perform a union bound over a set of unknown identity. Simple workarounds to circumvent this issue are to impose a finite state space, or to assume prior knowledge over a finite superset of $U$. In this paper we opt for the first option. It remains an open and highly non-trivial question as to how (and whether) the framework can cope with an infinite state space.

### E.4 Effective Horizon of the AX Problem and its Dependency on $\varepsilon$

$\text{UcbExplore}$ [1] designs finite-horizon problems with horizon $H_{UCB} := [L + L^2 \varepsilon^{-1}]$ and outputs policies that reset every $H_{UCB}$ time steps. In the following we prove that the effective horizon of the AX problem actually scales as $O(\log(L\varepsilon^{-1})L)$, i.e., only logarithmically w.r.t. $\varepsilon^{-1}$. We begin by defining the concept of “resetting” policies as follows.

**Definition 11.** For any $\pi \in \Pi$ and horizon $H \geq 0$, we denote by $\pi^H$ the non-stationary policy that executes the actions prescribed by $\pi$ and performs the RESET action every $H$ steps, i.e.,

$$\pi^H_t(a|s) := \begin{cases} \text{RESET} & \text{if } t \equiv 0 \text{ (mod } H), \\ \pi(a|s) & \text{otherwise.} \end{cases}$$

We denote by $\Pi^H$ the set of such “resetting” policies.

The following lemma captures the effective horizon $H_{eff}$ of the problem, in the sense that restricting our attention to $\Pi^H(S^{-1})$ for $H \geq H_{eff}$ does not compromise the possibility of finding policies that achieve the performance required by $\text{AX}^*$ (and thus also by $\text{AX}_L$).
Lemma 17. For any \( \varepsilon \in (0, 1] \) and \( L \geq 1 \), whenever
\[
H \geq H_{\text{eff}} := 4(L + 1) \left\lceil \log \left( \frac{4(L + 1)}{\varepsilon} \right) \right\rceil,
\]
we have for any \( s^1 \in \mathcal{S}_L^+ \),
\[
\min_{\pi^H \in \Pi^H(\mathcal{S}_L^+)} v_{\pi^H}(s_0 \rightarrow s^1) \leq V_{\mathcal{S}_L^+}^*(s_0 \rightarrow s^1) + \varepsilon.
\]

Proof. Consider any goal state \( s^1 \in \mathcal{S}_L^+ \). Set \( \varepsilon' := \frac{\varepsilon}{2(L+1)} \leq \frac{1}{2} \). Denote by \( \pi \in \Pi(\mathcal{S}_L^+) \) the minimizer of \( V_{\mathcal{S}_L^+}^*(s_0 \rightarrow s^1) \). For any horizon \( H \geq 0 \), we introduce the truncated value function
\[
v_{\pi,H}(s \rightarrow s') := \mathbb{E}[\tau_{\pi}(s \rightarrow s') \wedge H] \text{ and the tail probability } q_{\pi,H}(s \rightarrow s') := \mathbb{P}(\tau_{\pi}(s \rightarrow s') > H).
\]
Due to the presence of the RESET action, the value function of \( \pi \) can be bounded for all states
\[
s \in \mathcal{S}_L^+ \setminus \{s^1\} \text{ as }
\]
\[
v_{\pi}(s \rightarrow s^1) \leq V_{\mathcal{S}_L^+}^*(s_0 \rightarrow s^1) + 1 \leq L + 1.
\]
This entails that the probability of the goal-reaching time decays exponentially. More specifically, we have
\[
q_{\pi,H}(s_0 \rightarrow s^1) \leq 2 \exp \left( -\frac{H}{4(L + 1)} \right) \leq \varepsilon', \tag{21}
\]
where the first inequality stems from Lem. 16 and the second inequality comes from the choice of \( H \geq 4(L + 1) \left\lceil \log \left( \frac{2}{\varepsilon} \right) \right\rceil \). Furthermore, we have \( \tau_{\pi}(s \rightarrow s') \wedge H \leq \tau_{\pi}(s \rightarrow s') \) and thus
\[
\mathbb{E}[\tau_{\pi}(s \rightarrow s') \wedge H] \leq \mathbb{E}[\tau_{\pi}(s \rightarrow s')].
\]
Consequently,
\[
v_{\pi,H}(s_0 \rightarrow s^1) \leq v_{\pi}(s_0 \rightarrow s^1) = V_{\mathcal{S}_L^+}^*(s_0 \rightarrow s^1). \tag{22}
\]
Now, from [1, Eq. 4], the value function of \( \pi \) can be related to its truncated value function and tail probability as follows
\[
v_{\pi,H} = v_{\pi,H} + q_{\pi,H}.
\]
Plugging Eq. 21 and 22 into Eq. 23 yields
\[
v_{\pi,H}(s_0 \rightarrow s^1) \leq \frac{V_{\mathcal{S}_L^+}^*(s_0 \rightarrow s^1) + \varepsilon'}{1 - \varepsilon'}.
\]
Notice that the inequalities \( \frac{1}{1-x} \leq 1 + 2x \) and \( \frac{x}{1-x} \leq 2x \) hold for any \( 0 < x < \frac{1}{2} \). Applying them for \( x = \varepsilon' \) yields
\[
\frac{V_{\mathcal{S}_L^+}^*(s_0 \rightarrow s^1) + \varepsilon'}{1 - \varepsilon'} \leq (1 + 2\varepsilon')V_{\mathcal{S}_L^+}^*(s_0 \rightarrow s^1) + 2\varepsilon'.
\]
From the inequality \( V_{\mathcal{S}_L^+}^*(s_0 \rightarrow s^1) \leq L \) and the definition of \( \varepsilon' \), we finally obtain
\[
v_{\pi,H}(s_0 \rightarrow s^1) \leq V_{\mathcal{S}_L^+}^*(s_0 \rightarrow s^1) + \varepsilon,
\]
which completes the proof. \( \square \)

Lem. 17 reveals that the effective horizon \( H_{\text{eff}} \) of the AX problem scales only logarithmically and not linearly in \( \varepsilon^{-1} \). This highlights that the design choice in UcbExplore to tackle finite-horizon problems with horizon \( H_{\text{UCB}} \) unavoidably leads to a suboptimal dependency on \( \varepsilon \) in its AXC sample complexity bound. In contrast, by designing SSP problems and thus leveraging the intrinsic goal-oriented nature of the problem, DisCo can (implicitly) capture the effective horizon of the problem. This observation is at the heart of the improvement in the \( \varepsilon \) dependency from \( \tilde{O}(\varepsilon^{-3}) \) of UcbExplore [1] to \( \tilde{O}(\varepsilon^{-2}) \) of DisCo (Thm. 1).
F. Experiments

This section complements the experimental findings partially reported in Sect. 5. We provide details about the algorithmic configurations and the environments as well as additional experiments.

F.1 Algorithmic Configurations

Experimental improvements to UcbExplore [1]. We introduce several modifications to UcbExplore in order to boost its practical performance. We remove all the constants and logarithmic terms from the requirement for state discovery and policy evaluation (refer to [1, Fig. 1]). Furthermore, we remove the constants in the definition of the accuracy \( \epsilon' = \epsilon / L \) used by UcbExplore (while their original algorithm requires \( \epsilon' \) to be divided by 8, we remove this constant). We also significantly improve the planning phase of UcbExplore [1, Fig. 2]. Their procedure requires to divide the samples into \( H := (1 + 1/\epsilon')L \) disjoint sets to estimate the transition probability of each stage \( h \) of the finite-horizon MDP. This substantially reduces the accuracy of the estimated transition probability since for each stage \( h \) only \( N_k(s,a) / H \) are used. In our experiments, we use all the samples to estimate a stationary MDP (i.e., \( \hat{p}_k(s'|s,a) = N_k(s,a,s') / N_k(s,a) \)) rather than a stage-dependent model. Estimating a stationary model instead of bucketing the data is simpler and more efficient since leads to a higher accuracy of the estimated model. To avoid to move far away from the original UcbExplore, we decided to define the confidence intervals as if bucketing was used. We thus consider \( \hat{N}_k(s,a) = N_k(s,a) / H \) for the construction of the confidence intervals. For planning, we use the optimistic backward induction procedure as in [30]. We thus leverage empirical Bernstein inequalities —which are much tighter— rather than Hoeffding inequalities as suggested in [1]. In particular, we further approximate the bonus suggested in [30, Alg. 4] as

\[
\begin{align*}
    b_k(s,a) &= \sqrt{\frac{\text{Var}_{s'|\sim \hat{p}_k(s,a)}[N_{k,h+1}(s')]}{N_k(s,a) \vee 1}} + \frac{(H-h)}{N_k(s,a) \vee 1}.
\end{align*}
\]

For DisCo, we follow the same approach of removing constants and logarithmic terms. We thus use the definition of \( \phi \) as in Thm. 1 with \( \alpha = 1 \) and without log-terms. For planning, we use the procedure described in App. D with \( b_k(s,a,s') = \sqrt{\frac{\text{Var}_{s'|\sim \hat{p}_k(s,a)}[N_{k,s,a}]}{N_k(s,a) \vee 1}} + \frac{1}{N_k(s,a) \vee 1} \). Finally, in the experiments we use a state-action dependent value \( \hat{\Theta}(s,a,K_k) = \left( \sum_{s' \in K_k} \sqrt{\hat{p}_k(s'|s,a)(1 - \hat{p}_k(s'|s,a))} \right)^2 \) instead of taking the maximum over \( (s,a) \).

Even though we boosted the practical performance of UcbExplore w.r.t. the original algorithm proposed in [1] (e.g., the use of Bernstein), we believe it makes the comparison between DisCo and UcbExplore as fair as possible.

F.2 Confusing Chain

The confusing chain environment referred to in Sect. 5 is constructed as follows. It is an MDP composed of an initial state \( s_0 \), a chain of length \( C \) (states are denoted by \( s_1, \ldots, s_C \)) and a set of \( K \) confusing states \( (s_{C+1}, \ldots, s_{C+K}) \). Two actions are available in each state. In state \( s_0 \), we have a forward action \( a_0 \) that moves to the chain with probability \( p_c \) (\( p(s_i | s_0, a_0) = p_c \) and \( p(s_0 | s_0, a_0) = 1 - p_c \)) and a confusing action that has uniform probability of reaching any confusing state (\( p(s_i | s_0, a_1) = 1 / K \) for any \( i \in \{ C+1, \ldots, C+K \} \)). In the confusing states, all actions move deterministically to the end of the chain (\( p(s_C | s_i, a) = 1 \) for any \( i \in \{ C+1, \ldots, C+K \} \) and \( a \)). In each state of the chain, there is a forward action \( a_0 \) that behaves as in \( s_0 \) (\( p(s_{\min(C,i+1)} | s_i, a_0) = p_c \) and \( p(s_i | s_i, a_0) = 1 - p_c \) for any \( i \in \{ 1, \ldots, C-1 \} \)) and a skip action \( a_1 \) that moves to \( m \) states ahead with probability \( p_{\text{skip}} \) (\( p(s_{\min(C,i+m)} | s_i, a_0) = p_{\text{skip}} \) and \( p(s_i | s_i, a_0) = 1 - p_{\text{skip}} \) for any \( i \in \{ 1, \ldots, C-1 \} \)). Finally, \( p(s_0 | s_0, a) = 1 \) for any action \( a \). In our experiments, we set \( m = 4, p_{\text{skip}} = 1/3, p_c = 1, C = 5, K = 6, L = 4.5 \).
We set the initial state to be at $s$. We consider the combination lock problem introduced in [31]. The domain is a stochastic chain with $S=6$ states and $A=2$ actions. In each state $s_k$, action right ($a_1$) is deterministic and leads to state $s_{k+1}$, while action left ($a_0$) moves to a state $s_{k-1}$ with probability proportional to $1/(k-l)$ (i.e., inversely proportional to the distance of the states). Formally, we have that

$$n(x_k, x_l) = \begin{cases} \frac{1}{x-l} & \text{if } l < k \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad p(x_l|x_k, a_0) = \frac{n(x_k, x_l)}{\sum_s n(x_k, s)}.$$

We set the initial state to be at $2/3$ of the chain, i.e., $s_0 = \left\lfloor \frac{2N}{3} \right\rfloor$. The actions in the end states are absorbing, i.e., $p(s_0|s_0, a_0) = 1$ and $p(s_N|s_{N-1}, a_1) = 1$, while the remaining actions behave normally. See Fig. 5 for an illustration of the domain.

| $\varepsilon$ | DisCo | UcbExplore-Bernstein |
|---------------|-------|---------------------|
| 0.1           | 374,263 (13,906) | 5,076,688 (92,643) |
| 0.2           | 105,569 (4,645)  | 636,580 (13,716)   |
| 0.4           | 29,160 (829)     | 108,894 (2,305)    |
| 0.6           | 15,349 (475)     | 40,538 (805)       |
| 0.8           | 9,891 (244)      | 21,270 (441)       |

Table 2: Sample complexity of DisCo and UcbExplore-Bernstein, on the confusing chain domain. Values are averaged over 50 runs and the 95%-confidence interval of the mean is reported in parenthesis.

| $\varepsilon$ | $s_0$ | $s_1$ | $s_2$ | $s_3$ | $s_4$ | $s_5$ |
|---------------|-------|-------|-------|-------|-------|-------|
| 0.1, 0.2      | 0     | 1     | 2     | 3     | 4     | 4     |
| 0.4           | 0     | 1     | 2     | 3     | 4     | 4.94  (0.04) |
| 0.6           | 0     | 1     | 2     | 3.36  (0.11) | 4 | 4.53  (0.07) |
| 0.8           | 0     | 1     | 2     | 3.38  (0.11) | 4.07 (0.07) | 4.53  (0.06) |

Table 3: Expected hitting time of state $s_i$ of the goal-oriented policy $\pi_{s_i}$ recovered by UcbExplore-Bernstein, on the confusing chain domain. DisCo recovers the optimal goal-oriented policy in all the runs and for all $\varepsilon$. The advantage of DisCo lies in its final policy consolidation step. Values are averaged over 50 runs and the 95%-confidence interval of the mean is reported in parenthesis (it is omitted when equal to 0). This shows that UcbExplore recovers the optimal goal-oriented policy in every run only for $\varepsilon$ equal to 0.1 and 0.2.

Sample complexity. We provide in Tab. 2 the sample complexity of the algorithms for varying values of $\varepsilon$. As mentioned in Sect. 5, DisCo outperforms UcbExplore for any value of $\varepsilon$, and increasingly so when $\varepsilon$ decreases. Fig. 7 complements Fig. 2 for additional values of $\varepsilon$.

Quality of goal-reaching policies. We now investigate the quality of the policies recovered by DisCo and UcbExplore. In particular, we show that DisCo is able to find the incrementally near-optimal shortest-path policies to any goal state, while UcbExplore may only recover sub-optimal policies. On the confusing chain domain, the intuition is that the set of confusing states makes $s_C$ reachable in just 2 steps but the confusing states are not in the controllable set and thus the algorithms are not able to recover the shortest-path policy to $s_C$. On the other hand, state $s_C$ is controllable through two policies: 1) the policies $\pi_1$ that takes always the forward action $a_0$ reaches $s_C$ in 5 steps; 2) the policy $\pi_2$ that takes the skip action $a_1$ in $s_1$ reaches $s_C$ in 4 steps. We observed empirically that DisCo always recovers policy $\pi_1$ (i.e., the fastest policy) while UcbExplore selects policy $\pi_2$ in several cases. This is highlighted in Tab. 3 where we report the expected hitting time of the policies recovered by the algorithms. This finding is not surprising since, as we explain in Sect. 4 and App. A, UcbExplore is designed to find policies reaching states in at most $L$ steps on average, yet it is not able to recover incrementally near-optimal shortest-path policies, as opposed to DisCo.

F.3 Combination Lock

We consider the combination lock problem introduced in [31]. The domain is a stochastic chain with $S=6$ states and $A=2$ actions. In each state $s_k$, action right ($a_1$) is deterministic and leads to state $s_{k+1}$, while action left ($a_0$) moves to a state $s_{k-1}$ with probability proportional to $1/(k-l)$ (i.e., inversely proportional to the distance of the states). Formally, we have that

$$n(x_k, x_l) = \begin{cases} \frac{1}{x-l} & \text{if } l < k \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad p(x_l|x_k, a_0) = \frac{n(x_k, x_l)}{\sum_s n(x_k, s)}.$$
Figure 5: Combination lock domain with $S = 6$ states. Expected hitting times from the initial state $s_3$ are $v_π(s_3 \rightarrow s) = (2.18, 1.91, 1.64, 0, 1, 2)$. Consider $L = 3$, the set of incrementally $L$-controllable states is $S^*_L = \{s_2, s_3, s_4, s_5\}$. The goal-oriented policy to reach $s_4$ and $s_5$ takes always the right action $a_1$, while the policy for $s_2$ always selects the left action $a_0$.

![Diagram of combination lock domain](image)

Figure 6: Proportion of the incrementally $L$-controllable states identified by DisCo and UcbExplore in the combination lock domain for $L = 2.7$ and $\varepsilon = 0.2$. Values are averaged over 20 runs.

**Sample complexity.** We evaluate the two algorithms DisCo and UcbExplore on the combination lock domain, for $\varepsilon = 0.2$ and $L = 2.7$. We further boost the empirical performance of UcbExplore by using $N$ instead of $N$ for the construction of the confidence intervals (i.e., we do not account for the data bucketing in [1], see App. F.1). To preserve the robustness of the algorithm, we use $\log(|K_k|^2)/\varepsilon^6$ episodes for UcbExplore’s policy evaluation phase (indeed we noticed that the removal of the logarithmic term here sometimes leads UcbExplore to miss some states in $S^*_L$ in this domain). For the same reason, in DisCo we use the value $\hat{\Theta}(K_k) = \max_{s,a} \hat{\Theta}(s,a,K_k)$ prescribed by the theoretical algorithm instead of the state-action dependent values used in the previous experiment. We average the experiments over 20 runs and obtain a sample complexity of 30, 117 (2, 087) for DisCo and 90, 232 (2, 592) for UcbExplore. Fig. 6 reports the proportion of incrementally $L$-controllable states identified by the algorithms as a function of time. We notice that once again DisCo clearly outperforms UcbExplore.
Figure 7: Proportion of the incrementally $L$-controllable states identified by DisCo and UcbExplore on the confusing chain domain for $L = 4.5$ and $\varepsilon \in \{0.1, 0.2, 0.4, 0.6, 0.8\}$. Values are averaged over 50 runs. UcbExplore uses Bernstein confidence intervals for planning.