Effective Reinforcement Learning Exploration in Stochastic Shortest Paths with Dead-Ends

Gustavo De Mari Pereira [1], **Leliane Nunes de Barros** [1]

[1]Institute of Mathematics, Statistics and Computer Science
Rua do Matao, 1010
Sao Paulo, SP 05508-090
gustavodmp@ime.usp.br

Abstract

Stochastic Shortest Path (SSP) problems form the basis for goal-oriented reinforcement learning. In an SSP with dead-ends, exploration becomes particularly challenging, as optimistic exploration strategies can actively favor dangerous regions, while undirected exploration may become exponentially inefficient. We introduce Lexicographical Exploration, a goal-reachability mechanism integrated into the model-based RL architecture Dual Dyna-Q. Our approach leverages a learned goal-reachability function to dynamically constrain exploration to actions that preserve a non-negligible probability of reaching the goal. We evaluate multiple exploration strategies, including ϵ-greedy, Boltzmann, UCB, and MBIE-EB, across nine SSPs with dead-ends. Results show that Dual Dyna-Q combined with Lexicographical Exploration consistently improves reliability and sample efficiency compared to standard baselines while mitigating unsafe policies.

Introduction

Goal-oriented reinforcement learning (RL) is a fundamental paradigm for tasks where an autonomous agent must reach a specific goal state while minimizing cumulative costs. These problems are commonly modeled as Stochastic Shortest Path (SSP) problems (Bertsekas and Tsitsiklis 1991), where the objective is to find a policy that reaches a goal state with minimum expected cost. Classical SSP theory typically assumes that the goal is reachable from every state. However, in many realistic domains, this assumption is violated by the presence of *dead-ends*: states from which the goal is irreversibly unreachable (Kolobov and Weld 2012).

Previous work in probabilistic planning introduced optimization criteria for SSPs with dead-ends, including MAXPROB (Kolobov and Weld 2012), Stochastic Safest and Shortest Path (S^3P) (Teichteil-Königsbuch 2012), and Maximum Probability Minimum Cost (MCMP) (Trevizan, Teichteil-Konigsbuch, and Thiebaux 2017). These criteria focus on learning a policy where the agent must first maximize the probability of reaching the goal and then minimize expected cost. Because planning methods assume access to a known environment model, they can optimize these objectives without requiring exploratory interaction with the environment. In RL, however, the transition dynamics are initially unknown and must be discovered through exploration. Consequently, exploration itself may significantly impact the primary lexicographical objective: an exploratory mistake can prevent the agent from reaching the goal.

This mismatch exposes fundamental limitations in classical RL exploration strategies when applied to SSPs with dead-ends. Directed exploration methods based on the principle of Optimism in the Face of Uncertainty (OFU), such as UCB (Auer, Cesa-Bianchi, and Fischer 2002) and MBIE-EB (Strehl and Littman 2008), are highly effective in standard MDPs. However, in environments with dead-ends, unconstrained optimism becomes actively dangerous. By assigning large exploration bonuses to uncertain state-action pairs, optimistic agents are systematically encouraged to seek out unknown regions that may conceal irreversible failures. We refer to this phenomenon as *deadly curiosity*. Conversely, undirected exploration methods such as ϵ-greedy and Boltzmann exploration rely on stochasticity to ensure environment coverage. As shown by Whitehead (1991), these methods can suffer from exponential sample complexity in environments where trajectories are more likely to drift away from the goal than toward it. In SSPs with dead-ends, such inefficient wandering substantially increases the probability of entering dead-end states, transforming theoretical inefficiency into practical mission failure.

To bridge this gap between planning and reinforcement learning, we propose Lexicographical Exploration, a goal-reachability discovery mechanism integrated into a model-based dual-optimization architecture called Dual Dyna-Q. Our approach continuously learns a goal-reachability function (Q^P) and uses it to dynamically constrain both exploration and action selection to actions that preserve a non-negligible probability of reaching the goal. By explicitly decoupling reachability from cost during discovery, the proposed mechanism aligns exploration with lexicographical planning objectives that prioritize safety before cost minimization. To analyze the impact of exploration strategies in SSPs with dead-ends, we systematically compare our framework against baselines such as Q-learning, Dyna-Q, and Finite-Penalty Q-learning (FP-QL) under a variety of classical exploration methods. Our results demonstrate that incorporating goal-reachability into exploration substantially improves reliability and sample efficiency in environments with dead-ends.

Background

This section establishes the theoretical foundations of reinforcement learning and formalizes the SSP framework alongside its dead-end extensions.

Markov Decision Processes and Reinforcement Learning

The standard framework for sequential decision-making is the **Markov Decision Process (MDP)** (Puterman 2014; Sutton and Barto 2018). The tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$ defines an MDP, where $\mathcal{S}$ is a finite set of states, $\mathcal{A}$ is a finite set of actions, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition probability function, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the reward function, and $\gamma \in [0, 1]$ is the discount factor. In reinforcement learning (RL), the agent lacks prior knowledge of $\mathcal{T}$ and $\mathcal{R}$ and must learn an optimal policy through trial-and-error interaction.

Throughout this paper, we adopt the **cost-minimization** convention prevalent in the planning and SSP literature. Thus, we represent the agent's objective using a cost function $\mathcal{C}$ (where $\mathcal{C} = -\mathcal{R}$) and seek to minimize expected cumulative costs. The optimal value functions V^* and Q^* satisfy the **Bellman optimality equations** (Bellman 1957; Bertsekas 2017):

$$Q^*(s, a) = \sum_{s' \in \mathcal{S}} \mathcal{T}(s, a, s')[\mathcal{C}(s, a, s') + \gamma \min_{a' \in \mathcal{A}} Q^*(s', a')]. \tag{1}$$

The optimal value at state s is then $V^*(s) = \min_a Q^*(s, a)$. During learning, algorithms like Q-learning (Watkins and Dayan 1992) approximate these values using Temporal Difference (TD) learning (Sutton 1988) to iteratively update estimates from observed transitions (s, a, c, s').

Reinforcement learning algorithms fall into two broad paradigms: model-free and model-based. **Model-free** algorithms, such as Q-learning, learn an optimal policy directly from interactions with the environment without explicitly representing the underlying transition or reward dynamics. In contrast, **model-based** algorithms, such as Dyna-Q, seek to learn an internal model of the environment's dynamics ($\hat{\mathcal{T}}$) and rewards ($\hat{\mathcal{R}}$) from observed transitions. The agent can then use this model for **planning**, simulating experience to improve its value function and policy without further real-world interaction (Sutton and Barto 2018; Moerland et al. 2023).

Definition 1 (Policy) *A **stationary policy** $\pi : \mathcal{S} \to \mathcal{A}$ is a mapping from states to actions defining the agent's behavior.*

Definition 2 (Return) *The **return** G_t at time t is the cumulative cost accumulated by following a policy: $G_t = \sum_{k=0}^{\infty} \gamma^k \mathcal{C}(s_{t+k}, a_{t+k}, s_{t+k+1})$.*

Definition 3 (Value Functions) *The **value function** $V^\pi(s) = \mathbb{E}_\pi[G_t \mid s_t = s]$ represents the expected return from state s following policy π. Similarly, the **state-action value function** $Q^\pi(s, a) = \mathbb{E}_\pi[G_t \mid s_t = s, a_t = a]$ represents the expected return after taking action a in state s. An optimal policy π^* minimizes these expected costs for all states.*

Stochastic Shortest Path Problems

SSPs are a powerful superclass of standard MDPs specifically tailored for goal-oriented problems where the objective is to reach a terminal state with minimum cost.

Definition 4 (Stochastic Shortest Path (SSP)) *The tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{C}, s_0, \mathcal{G})$ defines an SSP, where $\mathcal{S}, \mathcal{A}, \mathcal{T}$ are as defined above, s_0 is the initial state, and $\mathcal{G} \subseteq \mathcal{S}$ is the set of goal states. For all $s_g \in \mathcal{G}$ and $a \in \mathcal{A}$, $\mathcal{T}(s_g, a, s_g) = 1$ and $\mathcal{C}(s_g, a, s') = 0$ for all $s' \in \mathcal{S}$. In standard SSPs, costs are undiscounted ($\gamma = 1$).*

Standard SSPs rely on two key assumptions (Bertsekas and Tsitsiklis 1991):

Assumption 1 *There exists at least one proper policy.*

Assumption 2 *Every improper policy incurs infinite expected cost for at least one state.*

Definition 5 (Proper Policy) *A policy π is **proper** if it reaches the goal with probability 1 from any state $s \in \mathcal{S}$, and **improper** otherwise (Bertsekas 2022).*

Kolobov and Weld (2012) and Kolobov (2013) show that we can compile any Infinite-Horizon Discounted MDP into an equivalent SSP by introducing a special goal state s_g and scaling the transition dynamics by the discount factor γ. This mapping preserves policy optimality and implies that the SSP framework is strictly more expressive than discounted MDPs. We provide the precise mathematical construction of this transformation in Appendix . This mapping allows us to treat both Finite-Horizon and Infinite-Horizon MDPs as specialized subclasses of the SSP model (Mausam and Kolobov 2012).

Definition 6 (SSP with Avoidable Dead-Ends (SSPADE)) *An SSPADE relaxes Assumption 1 by only requiring a proper policy to exist from the initial state s_0. Formally, there exists at least one policy π such that $P(\text{reach } \mathcal{G} | s_0, \pi) = 1$.*

Definition 7 (SSP with Unavoidable Dead-Ends (SSPUDE)) *The tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{C}, s_0, \mathcal{G}, D)$ defines an SSPUDE, where $D \in \mathbb{R}^+ \cup \{+\infty\}$ is a penalty for failing to reach the goal. In an SSPUDE, there is no proper policy from s_0.*

Depending on the value of D, we distinguish two cases: (1) **Finite-Penalty SSPUDE (fSSPUDE)**, where $D < \infty$ and the problem reduces to a standard SSP with a fixed penalty for dead-ends; and (2) **Infinite-Penalty SSPUDE (iSSPUDE)**, where $D = \infty$, requiring a lexicographical approach that first maximizes goal-reachability (MAXPROB) before minimizing cost.

Definition 8 (Trap) *A **trap** is a set of states $\Theta \subseteq \mathcal{S} \setminus \mathcal{G}$ such that for a given policy π, the agent remains within Θ with probability 1: $\sum_{s' \in \Theta} \mathcal{T}(s, \pi(s), s') = 1$ for all $s \in \Theta$. Any policy that enters a trap is necessarily improper, as the goal becomes unreachable.*

While discounting ($\gamma < 1$) prevents infinite costs in SSPs with dead-ends, it may distort the goal-oriented objective (Bertsekas 2023). However, it provides regularization by bounding self-loop costs to a finite limit. Specifically, for

a state s trapped in an infinite cycle or self-loop with a constant step cost $c > 0$, the cumulative cost is the infinite series $V(s) = \sum_{k=0}^{\infty} \gamma^k c = c \sum_{k=0}^{\infty} \gamma^k$. Under the discount factor $0 \leq \gamma < 1$, this geometric series converges to $\frac{1}{1-\gamma}$, resulting in the bounded value $V(s) = \frac{c}{1-\gamma}$. This convergence ensures numerical stability in value updates by preventing divergence to infinity, while automatically penalizing traps and bypassing more complex trap elimination mechanisms like **Find, Revise, and Eliminate Traps (FRET)** (Kolobov et al. 2011; Steinmetz, Hoffmann, and Buffet 2016).

Exploration in Reinforcement Learning

A central challenge in RL is the **exploration-exploitation trade-off**: the agent must balance between exploiting its current knowledge to minimize costs and exploring the environment to discover better policies. Thrun (1992) categorizes exploration strategies into two broad families: undirected and directed exploration.

Undirected exploration techniques rely on randomness to ensure environment coverage without utilizing knowledge about the learning process. Common strategies include ϵ**-greedy** exploration, which selects the greedy action $\arg\min_{a'} Q(s, a')$ with probability $1 - \epsilon + \frac{\epsilon}{|\mathcal{A}|}$ (and other actions with probability $\frac{\epsilon}{|\mathcal{A}|}$), and **Boltzmann (softmax)** exploration, which samples actions according to $P(a \mid s) = e^{Q(s,a)/\tau} / \sum_{a'} e^{Q(s,a')/\tau}$ under a temperature parameter $\tau > 0$. To ensure convergence, algorithms decay these parameters over time to satisfy the **Greedy in the Limit with Infinite Exploration (GLIE)** conditions (Singh et al. 2000).

Directed exploration techniques try to improve exploration efficiency by utilizing task-specific knowledge to guide the agent toward poorly understood regions. Modern directed techniques often formalize these heuristics using the principle of Optimism in the Face of Uncertainty (OFU). This principle is central to the Probably Approximately Correct in Markov Decision Processes (PAC-MDP) framework (Strehl and Littman 2008), which ensures that the agent learns a near-optimal policy with polynomial sample complexity.

Methods like Interval Estimation (IE) (Kaelbling 1993), **Upper Confidence Bounds (UCB)** (Auer, Cesa-Bianchi, and Fischer 2002), and **MBIE with Exploration Bonus (MBIE-EB)** (Strehl and Littman 2008) maintain uncertainty estimates or confidence intervals for the environment's dynamics. By adding an **exploration bonus** $b(s, a)$ to the reward function (or subtracting it from the cost), these algorithms incentivize discovery.

In practice, algorithms use the bonus for **action selection**, such as in UCB where $a_t = \arg\min_{a \in \mathcal{A}} [Q(s_t, a) - b(s_t, a)]$. Here, UCB uses the exploration bonus $b_{\text{UCB}}(s, a) = \beta \sqrt{\frac{2 \ln(1/\delta)}{N(s,a)}}$, while model-based methods like MBIE-EB employ $b_{\text{MBIE-EB}}(s, a) = \frac{\beta}{\sqrt{N(s,a)}}$.

Exploration for SSPs with Dead-Ends

The classic exploration-exploitation trade-off in reinforcement learning assumes the agent can freely probe the environment and recover from suboptimal decisions. In SSPs without dead-ends, all exploration is reversible: finite cost is incurred, but the goal remains reachable. However, in SSPs with dead-ends, this assumption breaks down. The presence of absorbing, zero-probability-of-success states introduces a critical safety constraint: the agent can no longer afford to explore arbitrarily, as doing so risks permanent failure. Consequently, exploration is no longer just about balancing efficiency and novelty, but about ensuring survival. This section details the fundamental challenges of exploration under such risk and introduces a goal-reachability-constrained framework to resolve this tension.

Challenges of Exploration with Dead-Ends

Dead-ends and Traps. The presence of dead-ends (states from which reaching the goal becomes impossible) severely complicates the exploration-exploitation dilemma. In SSPs with dead-ends, undirected exploration (such as ϵ-greedy or random walks) is dangerous; if an agent explores too freely, it risks executing actions that lead to catastrophic failures or unavoidable dead-ends. Once in a dead-end, the agent cannot reach the goal and may continue trapped to accumulate step costs indefinitely.

Sample Complexity and Convergence. This structural challenge has severe implications for **sample complexity** and algorithmic convergence. Unmodified algorithms, such as standard Value Iteration or simple ϵ-greedy strategies, cannot effectively evaluate the infinite costs associated with dead-ends. As a result, they waste resources exploring doomed states and fail to achieve polynomial sample complexity. Furthermore, the presence of dead-ends violates the foundational assumption of standard SSP theory: that a proper policy exists and can reach the goal with probability 1. When an agent executes a policy that enters a dead-end, it incurs an infinite expected cost, which directly leads to unbounded regret and prevents the Bellman equations from converging to a finite solution.

Exponential Complexity and Whitehead's Theorem. Whitehead's Theorem formalizes the danger of unguided discovery in such environments (Whitehead 1991; Thrun 1992). Whitehead established that undirected exploration scales exponentially with state space size if actions are more likely to lead away from the goal than toward it. In the context of SSPUDEs, this "drifting away" does not merely result in exponential delay but leads the agent into absorbing dead-ends, transforming theoretical complexity traps into practical mission failure. To prevent this, Lexicographical Exploration dynamically prunes the action space, restricting the agent to a safe region where the goal remains attainable. By prioritizing the identification of this safe subset over global coverage, the agent transforms a potentially divergent discovery process into a tractable, safety-first learning task.

Goal Reachability and Model Uncertainty. This goal-reachability-aware discovery is particularly critical in

model-based architectures, where exploration serves as the primary source of data for learning the empirical transition dynamics ($\hat{\mathcal{T}}$) and step costs ($\hat{\mathcal{C}}$). In standard model-learning, unguided exploration for maximum **state-space coverage** can waste samples in the learned model with trajectories that repeatedly go into dead-ends. High **model uncertainty** in these risk-prone regions is especially problematic during value propagation. In Dyna-style planning, if the agent propagates values from poorly understood traps into the safe regions of the state space, the resulting noise can corrupt the entire cost function Q^C, leading to sub-optimal or even risk-seeking policies. Lexicographical Exploration addresses this by ensuring that both model-learning and background planning remain focused on the safe subset, effectively shielding the value propagation process from the instability of dead-end regions.

Dual-Objective Model-Based Architecture

To address these challenges, the proposed framework decouples the problems of safety and efficiency by maintaining two distinct value functions: (1) a goal-reachability function $Q^P(s, a)$ estimating the maximum probability of reaching the goal; and (2) a cost function $Q^C(s, a)$ estimating the expected trajectory cost. By leveraging a model-based architecture (Dual Dyna-Q), the agent maintains empirical estimates of the transition dynamics ($\hat{\mathcal{T}}$) and step costs ($\hat{\mathcal{C}}$), providing a foundation for background planning across both objectives.

By explicitly learning and planning over the goal-reachability and cost objectives concurrently, the agent can coordinate sub-plans across different branches of the state space. This model-based coordination ensures that the overall policy prioritizes goal-reachability constraints (Q^P) before optimizing for cost (Q^C). Specifically, our cost update for Q^C minimizes expected cost restricted to the goal-reachability-optimal action set $\mathcal{A}^{MP}(s)$, providing an online realization of the S^3P criterion (Teichteil-Königsbuch 2012).

Lexicographical Exploration and Action Filtering

We propose Lexicographical Exploration, a goal-reachability-filtered discovery mechanism that prevents "deadly curiosity". To drive discovery, we optimistically initialize the goal-reachability function Q^P to 1.0, ensuring the agent initially explores all state-action pairs before observed failures drive estimates towards zero. Our framework utilizes the learned goal-reachability value function $Q^P(s, a)$ to define a set of **safe actions** $\mathcal{A}^{MP}(s)$ to which both exploration and exploitation are restricted:

$$\mathcal{A}^{MP}(s) = \{a \in \mathcal{A}(s) \mid Q^P(s, a) \geq \max_{a'} Q^P(s, a') - \epsilon_P\}, \tag{2}$$

where $\epsilon_P \geq 0$ is a goal-reachability tolerance. Our framework restricts both exploitation and random exploration to $\mathcal{A}^{MP}(s)$, effectively suppressing the discovery drive in regions where the agent estimates the goal is unreachable.

In our framework, the goal-reachability head acts as a learned **numeric nogood** (Kolobov and Weld 2012; Kolobov 2013). While classical nogoods in deterministic planning explicitly prune states that cannot reach the goal, our lexicographical filter provides a soft, learned approximation that adapts as the agent gains more evidence. The reachability tolerance ϵ_P plays a crucial role in this adaptation; it ensures that near-optimal safe actions are not prematurely pruned due to minor fluctuations in Q^P during the early stages of learning or in regions with high transition stochasticity. This parameter effectively governs the agent's "goal-reachability risk": a larger ϵ_P allows for more speculative discovery at the cost of safety, while a smaller value prioritizes survival but risks the permanent pruning of valid paths if initial estimates are pessimistic. Crucially, the model-based architecture of Dual Dyna-Q provides a robust mechanism for recovery from such errors. If a safe path is mistakenly pruned, background planning over updated transition counts eventually propagates higher goal-reachability values from discovered goal trajectories back to the decision point, potentially raising Q^P above the threshold and "re-opening" the path for exploration. Conversely, if the agent mistakenly identifies a dead-end as safe, the subsequent failure transitions observed during discovery will rapidly drive Q^P towards zero, allowing the lexicographical filter to prune the unsafe action in future episodes.

Beyond safety, this approach of maximizing goal probability while reducing costs via lexicographical exploration provides an advantage in computational and sample efficiency. By restricting the set of available actions to $\mathcal{A}^{MP}(s)$, the agent effectively prunes the environment's search tree, notably decreasing the search space of the exploration process. This reduction allows the agent to bypass the exponential complexity traps that Whitehead (1991) originally warned about in undirected discovery. Instead of wasting environmental samples and computational resources updating parameters for sub-optimal or doomed actions, the agent concentrates its experience on the reachable portion of the state space. This prioritization directly aligns with the dual-optimization criteria from planning, such as S^3P (Teichteil-Königsbuch 2012) and MCMP (Trevizan, Teichteil-Konigsbuch, and Thiebaux 2017), where action filtering ensures that we only consider maximal-probability policies. By avoiding the accumulation of experiences in unsafe areas that contain dead-ends, we accelerate the learning

Algorithm 1 Lexicographical Exploration

Require: State s, reachability function Q^P, cost function Q^C, tolerance ϵ_P, exploration rate ϵ
Ensure: Selected action a_t
1: Calculate maximal reachability: $P_{max} \leftarrow \max_{a' \in \mathcal{A}(s)} Q^P(s, a')$
2: Identify safe action set:
    $\mathcal{A}^{MP}(s) \leftarrow \{a \in \mathcal{A}(s) \mid Q^P(s, a) \geq P_{max} - \epsilon_P\}$
3: With probability ϵ:
    $a_t \leftarrow$ sample uniformly from $\mathcal{A}^{MP}(s)$
4: Otherwise:
    $a_t \leftarrow \arg\min_{a \in \mathcal{A}^{MP}(s)} Q^C(s, a)$
5: **return** a_t

process, transforming a potentially exponential search into a tractable, goal-reachability-aware discovery task.

Algorithm 2 Dual Dyna-Q with Lexicographical Exploration

Input: Learning rate α, discount factor γ, planning steps K, tolerance ϵ_P, min samples M
Initialize: $Q^P(s,a) \leftarrow 1$, $(Q^P(s_g,a)=1)$, $Q^C(s,a) \leftarrow 0$, Model $\mathcal{M} \leftarrow \emptyset$
for each episode **do**
  $s \leftarrow s_0$
  while $s \notin \mathcal{G}$ and not terminated **do**
    $a \leftarrow$ LEXICOGRAPHICALEXPLORATION$(s, Q^P, Q^C, \epsilon_P)$ ▷ Alg. 1
    Execute a, observe cost c and next state s'
    Update Model $\mathcal{M}(s,a) \leftarrow (c, s')$
    MAXPROBUPDATE(s,a)
    COSTUPDATE(s,a)
    for $k = 1$ **to** K **do** ▷ Planning Phase
      $(s_p, a_p) \leftarrow$ SAMPLEFROM$(\mathcal{M})$
      MAXPROBUPDATE(s_p, a_p)
      COSTUPDATE(s_p, a_p)
    end for
    $s \leftarrow s'$
  end while
end for

procedure MAXPROBUPDATE(s,a)
  $\hat{\mathcal{T}}(s,a,s') = \frac{N(s,a,s')}{\sum_{s''} N(s,a,s'')}$
  $V^P(s') = \begin{cases} 1 & \text{if } s' \in \mathcal{G} \\ \max_{a'} Q^P(s',a') & \text{otherwise} \end{cases}$
  $Q^P(s,a) \leftarrow (1-\alpha)Q^P(s,a) + \alpha \sum_{s'} \hat{\mathcal{T}}(s,a,s') \cdot \gamma \cdot V^P(s')$
end procedure

procedure COSTUPDATE(s,a)
  $\hat{\mathcal{T}}(s,a,s') = \frac{N(s,a,s')}{\sum_{s''} N(s,a,s'')}$
  if $\sum_{s'} N(s,a,s') \geq M$ **and** $Q^P(s,a) > 10^{-4}$ **then**
    $V^C(s') \leftarrow \min_{a' \in \mathcal{A}^{MP}(s')} Q^C(s',a')$
    $\bar{Q}^C \leftarrow \sum_{s'} \hat{\mathcal{T}}(s,a,s') V^P(s') \left[\hat{\mathcal{C}}(s,a,s') + \gamma V^C(s')\right]$
    $Q^C(s,a) \leftarrow \bar{Q}^C / Q^P(s,a)$
  else
    if $Q^P(s,a) < 10^{-4}$ **then**
      $Q^C(s,a) \leftarrow 0$ ▷ Stability gate
    end if
  end if
end procedure

Furthermore, this lexicographical approach provides a form of structural bias for discovery. Recent work on logic-driven directed exploration (Bagatella, Krause, and Martius 2025) highlights how utilizing high-level task specifications can densify the feedback signal during discovery. In our framework, the Q^P function serves a similar purpose: it transforms the sparse global objective of reaching the goal into a dense, local goal-reachability signal. This allows the agent to distinguish between "productive exploration", discovery that maintains or improves the likelihood of success, and "destructive curiosity", which leads to failure. By filtering the exploration pool, Dual Dyna-Q concentrates its sample budget on the reachable region of the state space, anchoring both model-learning and value propagation to the goal-oriented task specification.

Experiments

In this section, we empirically evaluate the proposed exploration framework across nine diverse SSP benchmarks with dead-ends. Our goal is to assess the impact of different exploration strategies in SSPs with dead-ends.

Experimental Setup

We evaluate our algorithms against standard Q-learning and Dyna-Q baselines. We use a tabular setting to isolate the optimization criteria, avoiding the confounding implementation heuristics (Engstrom et al. 2020) and the "deadly triad" of instability (Sutton and Barto 2018; van Hasselt et al. 2018) common in deep RL. This ensures that observed performance differences stem directly from our optimization objectives rather than architectural sensitivities (Henderson et al. 2018).

Benchmark Environments.

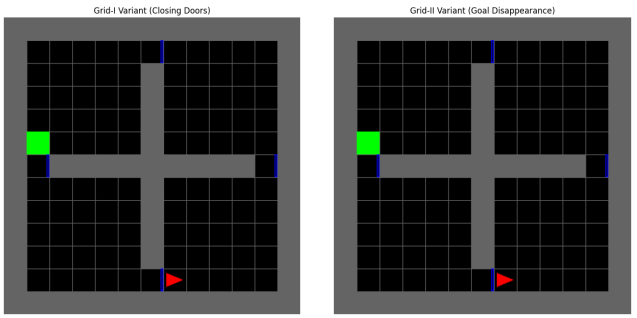
Figure 1 illustrates the layouts of the primary environments used in our experiments, including the custom grid worlds (Teichteil-Königsbuch 2012), navigation domains, and Probabilistic PDDL (Younes and Littman 2004) domains from the International Planning Competition (IPC) (Younes et al. 2005; Little and Thiebaux 2007; Bryce and Buffet 2008a) that contain dead-ends and complex choices between goal-reachability and costs. More details of each environment are provided in the Appendix.

Evaluation and Validation Metrics

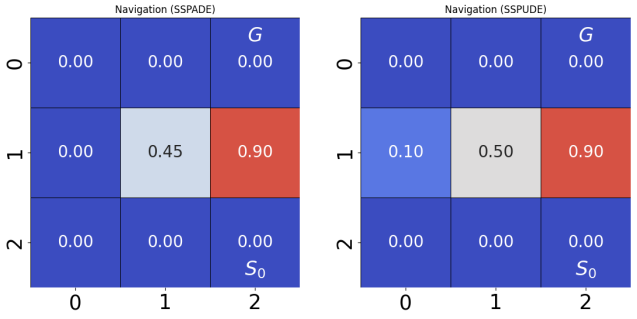
All experiments are conducted over 30 independent seeds with deterministic seeding to ensure reproducibility, using the algorithms **Dual Dyna-Q**, **Dyna-Q**, **Finite Penalty Q-learning**, and standard **Q-learning**. We compare standard exploration strategies against our proposed Lexicographical Exploration within the Dual Dyna-Q framework. Each agent is trained for 100,000 timesteps. Every 5,000 steps, we evaluate the agent over 100 greedy episodes ($\epsilon = 0$), reporting the mean and standard deviation for:

1. **Success Rate:** the empirical probability of reaching a goal state within the step limit without entering a dead-end;
2. **Success Cost:** the average cumulative cost during successful trajectories;
3. **Mean Cost:** the average undiscounted cumulative costs across all episodes, penalizing resource waste on failures.

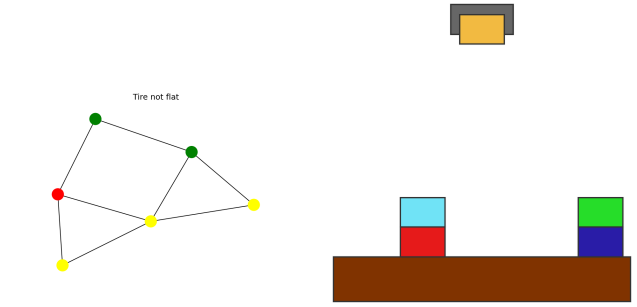
Additionally, we evaluate the exploration efficiency by measuring the state-space coverage during training, tracking both the unique visited states and unique state-action transitions. The corresponding results and learning curves are detailed in the Appendix (see Tables 6 and 7).



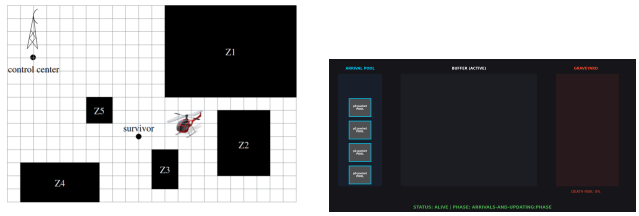
(a) Grid-I (b) Grid-II



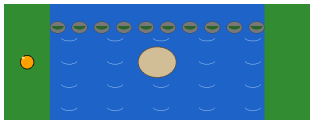
(c) Navigation SSPADE (d) Navigation SSPUDE



(e) Triangle Tireworld (f) Exploding Blocksworld



(g) Search-and-Rescue (h) Schedule



(i) River

Figure 1: Visual representations of the benchmark environments

Results and Discussion

The performance of the evaluated exploration strategies across all benchmark environments is summarized in Tables 1, 2, and 3. Our results demonstrate that the Lexicographical Exploration framework consistently provides the best balance between goal attainment and discovery efficiency across both navigation and symbolic planning tasks.

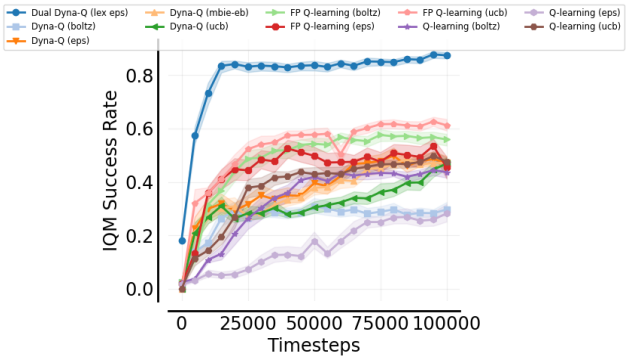


Figure 2: Success Rate Sample Efficiency

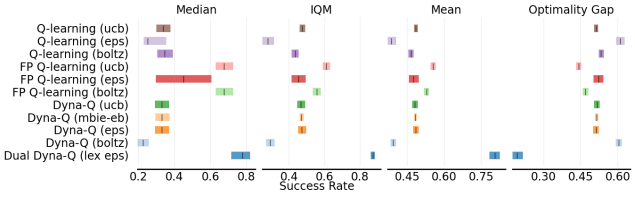


Figure 3: Aggregate Success Rate Metrics

Aggregate Statistical Evaluation. To ensure the reliability of our findings across diverse benchmarks, we present aggregate statistical analysis following the recommendations of Agarwal et al. (2021). The success rate sample efficiency curve (Fig 2) demonstrates that Dual Dyna-Q achieves faster and more stable goal-reachability discovery than all baselines, reaching an Interquartile Mean (IQM) above 0.8 within the first 15,000 training steps (15% of the total budget). In contrast, standard Q-learning baselines struggle to exceed an IQM of 0.3 even after 100,000 steps. The aggregate metrics summarize the final performance, showing that Dual Dyna-Q results in higher median (0.78), IQM (0.87), and mean (0.81) success rates. Crucially, while all standard baselines suffer from complete failure (achieving 0% success rate) in at least one environment, Dual Dyna-Q with Lexicographical Exploration avoids this by pruning unsafe regions, leveraging more efficient exploration to obtain a robust, non-zero success rate across all domains. Our approach results in an improvement over the best model-free baseline (FP-QL with UCB) by more than 35 percentage points in IQM. Furthermore, the success rate gap relative to an ideal success rate of 1.0 (reported as the Optimality Gap) is minimized to approximately 0.19 for Dual Dyna-Q. Since a success rate of 1.0 is physically unreachable in environments

Table 1: Success rate at the end of training for all benchmark environments (mean $\pm$ standard deviation).

Algorithm	Exploration	Environment								
		Exploding-Blocksworld	Grid-I	Grid-II	Navigation-SSPADE	Navigation-SSPUDE	River	Schedule	Search-and-Rescue	Triangle-Tireworld
Dual Dyna-Q	lex ϵ-greedy	0.643 ± 0.336	$\mathbf{0.545 \pm 0.110}$	$\mathbf{0.455 \pm 0.214}$	$\mathbf{1.000 \pm 0.000}$	0.870 ± 0.108	0.613 ± 0.078	0.999 ± 0.003	0.604 ± 0.128	$\mathbf{1.000 \pm 0.000}$
Dyna-Q	ϵ-greedy	$\mathbf{0.937 \pm 0.036}$	0.000 ± 0.000	0.000 ± 0.000	$\mathbf{1.000 \pm 0.000}$	0.061 ± 0.232	0.493 ± 0.048	0.999 ± 0.004	0.711 ± 0.062	0.494 ± 0.056
	Boltzmann	0.100 ± 0.194	0.000 ± 0.000	0.000 ± 0.000	$\mathbf{1.000 \pm 0.000}$	0.031 ± 0.168	0.506 ± 0.057	0.999 ± 0.004	0.732 ± 0.044	0.490 ± 0.056
	MBIE-EB	0.928 ± 0.035	0.000 ± 0.000	0.000 ± 0.000	$\mathbf{1.000 \pm 0.000}$	0.000 ± 0.000	0.493 ± 0.048	0.999 ± 0.003	$\mathbf{0.736 \pm 0.046}$	0.516 ± 0.050
	UCB	0.866 ± 0.124	0.000 ± 0.000	0.000 ± 0.000	$\mathbf{1.000 \pm 0.000}$	0.059 ± 0.225	0.493 ± 0.048	0.999 ± 0.003	0.725 ± 0.054	0.513 ± 0.047
FP Q-learning	ϵ-greedy	0.039 ± 0.111	0.000 ± 0.000	0.000 ± 0.000	$\mathbf{1.000 \pm 0.000}$	0.447 ± 0.455	$\mathbf{0.643 \pm 0.061}$	$\mathbf{1.000 \pm 0.000}$	0.107 ± 0.140	$\mathbf{1.000 \pm 0.000}$
	Boltzmann	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	$\mathbf{1.000 \pm 0.000}$	0.877 ± 0.168	0.642 ± 0.061	0.999 ± 0.003	0.180 ± 0.168	$\mathbf{1.000 \pm 0.000}$
	UCB	0.112 ± 0.188	0.000 ± 0.000	0.000 ± 0.000	$\mathbf{1.000 \pm 0.000}$	0.896 ± 0.033	0.642 ± 0.061	0.999 ± 0.003	0.275 ± 0.122	$\mathbf{1.000 \pm 0.000}$
Q-learning	ϵ-greedy	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	$\mathbf{1.000 \pm 0.000}$	0.240 ± 0.404	0.497 ± 0.055	$\mathbf{1.000 \pm 0.000}$	0.574 ± 0.058	0.508 ± 0.063
	Boltzmann	0.036 ± 0.110	0.000 ± 0.000	0.000 ± 0.000	$\mathbf{1.000 \pm 0.000}$	0.878 ± 0.169	0.500 ± 0.054	0.999 ± 0.003	0.605 ± 0.088	0.502 ± 0.053
	UCB	0.099 ± 0.117	0.000 ± 0.000	0.000 ± 0.000	$\mathbf{1.000 \pm 0.000}$	$\mathbf{0.906 \pm 0.026}$	0.496 ± 0.050	0.999 ± 0.003	0.700 ± 0.057	0.493 ± 0.057

Table 2: Success cost (undiscounted) at the end of training for all benchmark environments (mean $\pm$ standard deviation). Lower is better.

Algorithm	Exploration	Environment								
		Exploding-Blocksworld	Grid-I	Grid-II	Navigation-SSPADE	Navigation-SSPUDE	River	Schedule	Search-and-Rescue	Triangle-Tireworld
Dual Dyna-Q	lex ϵ-greedy	8.604 ± 3.593	$\mathbf{18.621 \pm 8.020}$	$\mathbf{19.680 \pm 8.158}$	$\mathbf{5.000 \pm 0.000}$	$\mathbf{4.867 \pm 0.507}$	0.423 ± 0.286	29.017 ± 2.767	10.771 ± 1.449	5.351 ± 0.243
Dyna-Q	ϵ-greedy	7.001 ± 0.008	–	–	$\mathbf{5.000 \pm 0.000}$	5.000 ± 0.000	$\mathbf{0.000 \pm 0.000}$	27.919 ± 2.509	6.647 ± 0.230	$\mathbf{1.000 \pm 0.000}$
	Boltzmann	7.744 ± 2.231	–	–	$\mathbf{5.000 \pm 0.000}$	5.000 ± 0.000	$\mathbf{0.000 \pm 0.000}$	28.836 ± 2.360	6.758 ± 0.140	$\mathbf{1.000 \pm 0.000}$
	MBIE-EB	$\mathbf{7.000 \pm 0.000}$	–	–	$\mathbf{5.000 \pm 0.000}$	–	$\mathbf{0.000 \pm 0.000}$	27.794 ± 2.335	6.670 ± 0.195	$\mathbf{1.000 \pm 0.000}$
	UCB	7.002 ± 0.009	–	–	$\mathbf{5.000 \pm 0.000}$	5.000 ± 0.000	$\mathbf{0.000 \pm 0.000}$	29.508 ± 2.943	6.747 ± 0.158	$\mathbf{1.000 \pm 0.000}$
FP Q-learning	ϵ-greedy	12.300 ± 9.315	–	–	$\mathbf{5.000 \pm 0.000}$	5.000 ± 0.000	0.624 ± 0.058	29.066 ± 1.920	12.227 ± 2.020	5.195 ± 0.208
	Boltzmann	–	–	–	$\mathbf{5.000 \pm 0.000}$	5.000 ± 0.000	0.619 ± 0.059	$\mathbf{27.712 \pm 3.056}$	11.729 ± 2.377	5.228 ± 0.209
	UCB	7.879 ± 2.000	–	–	$\mathbf{5.000 \pm 0.000}$	5.000 ± 0.000	0.619 ± 0.059	29.494 ± 2.537	11.709 ± 1.372	5.169 ± 0.242
Q-learning	ϵ-greedy	–	–	–	$\mathbf{5.000 \pm 0.000}$	5.000 ± 0.000	$\mathbf{0.000 \pm 0.000}$	30.371 ± 2.693	$\mathbf{6.305 \pm 0.494}$	$\mathbf{1.000 \pm 0.000}$
	Boltzmann	$\mathbf{7.000 \pm 0.000}$	–	–	$\mathbf{5.000 \pm 0.000}$	5.000 ± 0.000	$\mathbf{0.000 \pm 0.000}$	$\mathbf{27.712 \pm 3.056}$	6.378 ± 0.405	$\mathbf{1.000 \pm 0.000}$
	UCB	$\mathbf{7.000 \pm 0.000}$	–	–	$\mathbf{5.000 \pm 0.000}$	5.000 ± 0.000	$\mathbf{0.000 \pm 0.000}$	29.494 ± 2.537	6.561 ± 0.196	$\mathbf{1.000 \pm 0.000}$

Table 3: Mean cost (undiscounted) at the end of training for all benchmark environments (mean $\pm$ standard deviation). Lower is better.

Algorithm	Exploration	Environment								
		Exploding-Blocksworld	Grid-I	Grid-II	Navigation-SSPADE	Navigation-SSPUDE	River	Schedule	Search-and-Rescue	Triangle-Tireworld
Dual Dyna-Q	lex ϵ-greedy	28.063 ± 24.511	249.429 ± 87.128	264.320 ± 87.297	$\mathbf{5.000 \pm 0.000}$	30.350 ± 20.845	0.730 ± 0.177	29.187 ± 2.883	35.287 ± 23.815	5.351 ± 0.243
Dyna-Q	ϵ-greedy	$\mathbf{13.966 \pm 6.369}$	$\mathbf{200.000 \pm 0.000}$	$\mathbf{200.000 \pm 0.000}$	$\mathbf{5.000 \pm 0.000}$	188.105 ± 45.274	0.507 ± 0.048	28.090 ± 2.665	17.860 ± 12.696	$\mathbf{1.000 \pm 0.000}$
	Boltzmann	141.721 ± 44.897	201.533 ± 8.398	201.200 ± 6.573	$\mathbf{5.000 \pm 0.000}$	194.020 ± 32.754	$\mathbf{0.494 \pm 0.057}$	29.009 ± 2.305	$\mathbf{9.891 \pm 4.263}$	$\mathbf{1.000 \pm 0.000}$
	MBIE-EB	15.459 ± 6.978	$\mathbf{200.000 \pm 0.000}$	$\mathbf{200.000 \pm 0.000}$	$\mathbf{5.000 \pm 0.000}$	200.000 ± 0.000	0.507 ± 0.048	27.967 ± 2.336	13.187 ± 5.299	$\mathbf{1.000 \pm 0.000}$
	UCB	26.053 ± 22.477	202.000 ± 10.954	201.333 ± 7.303	$\mathbf{5.000 \pm 0.000}$	188.495 ± 43.790	0.507 ± 0.048	29.623 ± 2.880	10.489 ± 4.464	$\mathbf{1.000 \pm 0.000}$
FP Q-learning	ϵ-greedy	163.411 ± 40.363	224.616 ± 44.746	219.520 ± 36.384	$\mathbf{5.000 \pm 0.000}$	112.835 ± 88.717	0.855 ± 0.059	29.066 ± 1.920	158.508 ± 40.185	5.195 ± 0.208
	Boltzmann	171.752 ± 35.119	215.789 ± 35.473	215.032 ± 35.757	$\mathbf{5.000 \pm 0.000}$	29.050 ± 32.687	0.851 ± 0.061	$\mathbf{27.885 \pm 3.030}$	145.637 ± 38.944	5.228 ± 0.209
	UCB	143.022 ± 41.622	211.640 ± 26.441	213.051 ± 32.418	$\mathbf{5.000 \pm 0.000}$	25.215 ± 6.487	0.851 ± 0.061	29.608 ± 2.552	120.397 ± 24.534	5.169 ± 0.242
Q-learning	ϵ-greedy	184.656 ± 31.318	201.600 ± 8.764	201.733 ± 9.494	$\mathbf{5.000 \pm 0.000}$	153.265 ± 78.852	0.503 ± 0.055	30.371 ± 2.693	55.090 ± 15.501	$\mathbf{1.000 \pm 0.000}$
	Boltzmann	152.587 ± 41.168	207.600 ± 17.830	208.733 ± 20.168	$\mathbf{5.000 \pm 0.000}$	28.725 ± 32.962	0.500 ± 0.054	$\mathbf{27.885 \pm 3.030}$	44.014 ± 17.850	$\mathbf{1.000 \pm 0.000}$
	UCB	147.299 ± 41.130	204.533 ± 14.200	204.667 ± 14.756	$\mathbf{5.000 \pm 0.000}$	$\mathbf{23.395 \pm 4.983}$	0.504 ± 0.050	29.608 ± 2.552	19.711 ± 10.916	$\mathbf{1.000 \pm 0.000}$

with unavoidable dead-ends (where the optimal policy itself has a success rate strictly less than 1.0), the true gap to the optimal policy's performance is even smaller.

Success Rate Analysis. The empirical success rate (Table 1) serves as the primary metric for reliability in environments with dead-ends. In domains such as *Grid-I* and *Grid-II*, every standard optimistic and undirected baseline (UCB, MBIE-EB, Boltzmann) within both the Dyna-Q and Q-learning architectures exhibited a failure mode, achieving a 0% success rate. By assigning high curiosity bonuses or stochasticity to unvisited transitions that led directly into absorbing traps, these agents were disabled during the discovery phase. In contrast, the Dual Dyna-Q algorithm, leveraging Lexicographical Exploration, maintained a measurable reliability baseline, surviving in these environments with success rates of 55% in *Grid-I* and 46% in *Grid-II*. Furthermore, Dual Dyna-Q achieved high reliability in *Navigation-SSPUDE* (87%) and *Triangle-Tireworld* (100%). This confirms that decoupling safety (goal-reachability) from cost via lexicographical filtering is a fundamental architectural requirement for survival in environments with dead-ends.

Success Cost vs. Mean Undiscounted Cost. Our evaluation highlights a critical distinction between path efficiency (*Success Cost*) and overall resource waste (*Mean Undiscounted Cost*). As shown in Table 2, baselines seemingly achieve lower success costs in environments like *Triangle-Tireworld* (1.00 ± 0.00 vs. 5.35 ± 0.24 for Dual Dyna-Q). This apparent efficiency, however, is a misleading artifact of survival bias: a baseline only contributes to the success cost statistic when it takes the shortest, most dangerous path and survives purely by chance. These isolated "lucky" runs mask the agent's failure in the vast majority of episodes. Evaluating all attempts via Mean Undiscounted Cost (Table 3) reveals a more nuanced dynamic. In *Grid-I*, Dual Dyna-Q incurs a higher mean cost (249.4 ± 87.1) than model-free baselines like FP-QL (224.6 ± 44.7 or lower). This discrepancy occurs because baselines safely time out in the starting room to avoid the lockable doors entirely, whereas Dual Dyna-Q actively explores and occasionally triggers the dead-end penalty. Conversely, in *Navigation-SSPUDE*, Lexicographical Exploration achieves a mean cost of 30.4, while standard ϵ-greedy Q-learning incurs 153.3, both use the same base exploration strategy, isolating the effect of lexicographic fil-

tering. This sharp contrast accurately reflects the massive penalty standard agents accumulate through frequent failed attempts.

State and Action Space Exploration. State and transition counts (Appendix Tables 6 and 7) provide evidence about the lexicographical filter's pruning efficiency and positive impact on sample complexity. In complex environments, Dual Dyna-Q explores significantly smaller spaces than baselines. Dual Dyna-Q only appears to explore more in *Grid-I* (87.1 states, 428.9 transitions) than Dyna-Q Boltzmann (64.3, 303.7) because baselines fail to cross lockable doors and remain trapped in the start room. In the *Schedule* environment, Dual Dyna-Q explores just 198.0 ± 29.3 states and 320.5 ± 44.2 transitions, compared to standard Q-learning's 1123.7 ± 34.7 and 1809.0 ± 40.9. Standard exploration methods blindly traverse dangerous branches and waste sample budget in dead-ends, resulting in increased sample complexity. Conversely, Lexicographical Exploration uses the goal-reachability function Q^P to prune unsafe actions and avoid dead-ends.

Related Work

Our work builds upon foundational research in goal-oriented planning and reinforcement learning (Bellman 1957; Sutton and Barto 2018; Bertsekas 2017).

Probabilistic Planning and Dead-Ends. Automated planning traditionally assumes the goal is always reachable (Ghallab, Nau, and Traverso 2004). To address dead-ends in SSPs, Kolobov and Weld (2012) introduced the MAXPROB and Finite Penalty criteria, and subsequent work developed lexicographical frameworks such as S^3P (Teichteil-Königsbuch 2012) and MCMP (Trevizan, Teichteil-Konigsbuch, and Thiebaux 2017). While these methods focus on offline planning with known models, our work extends these criteria to online reinforcement learning with unknown transition dynamics.

Safe and Multi-Objective RL. Our lexicographical approach builds upon the foundations of Multi-Objective RL (MORL) (Hayes et al. 2023) and Thresholded Lexicographic Q-learning (TLQ-learning) (Vamplew et al. 2011). We adapt this paradigm to SSPUDEs, enforcing a hard goal-reachability constraint for survival before optimizing costs. This differs from standard Safe RL methods that often utilize soft penalties or Lagrangian multipliers (Altman 1999; Achiam et al. 2017; Garcıa and Fernandez 2015), prioritizing goal attainment as the primary discovery constraint (Fatemi and Sharma 2019).

Goal-Conditioned RL and Secure Exploration. While Goal-Conditioned RL often focuses on generalization across diverse reachable goals (Colas et al. 2022), our work addresses the reliability of discovery in the presence of unavoidable dead-ends. Our framework is closely related to **Secure Exploration** (Fatemi and Sharma 2019), which utilizes auxiliary value functions to protect agents from catastrophic failures. While their model-free approach focuses on security caps for random exploration, our Lexicographical Exploration leverages the learned model in Dual Dyna-Q to define a mathematically rigorous goal-reachability set, enabling the safe application of directed exploration strategies (such as UCB or MBIE-EB) in risk-prone environments. Similarly, while recent work explores no-regret exploration for goal-oriented MDPs (Tarbouriech 2022), they typically assume proper policies, a requirement our method relaxes.

Formal Methods and Model Checking. Probabilistic model checking provides safety assurances via exhaustive state-space analysis or AND-OR graph pruning (Kwiatkowska, Norman, and Parker 2022; Buffet 2007). Recent work also directs exploration using high-level LTL specifications to distill intrinsic rewards (Bagatella, Krause, and Martius 2025). While these formal approaches are highly reliable, they typically require complete models or known specifications and lack scalability in unknown domains. Conversely, our online RL method discovers cost-minimizing, goal-reaching policies through Lexicographical Exploration without explicitly constructing the full state space, bridging the gap between formal safety criteria and autonomous discovery.

Conclusion

This paper has addressed the challenge of exploration in SSP problems with dead-ends, where the principle of optimism in the face of uncertainty can lead to "deadly curiosity" and undirected exploration may be inefficient. We introduced Lexicographical Exploration, a goal-reachability-filtered discovery mechanism that dynamically prunes unsafe actions to ensure survival during the learning process. Our empirical evaluation across nine benchmarks shows that the Dual Dyna-Q framework with Lexicographical Exploration was the only method to achieve a positive success rate across all environments. Specifically, in environments such as Grid-I and Grid-II, standard methods achieve a 0% success rate, while our framework achieves success rates of 54.5% and 45.5%, respectively.

Our results indicate that decoupling safety (goal-reachability) from efficiency (cost) is an effective strategy for reinforcement learning algorithms in environments with dead-ends. By restricting both random exploration and greedy exploitation to a safe region where the goal remains attainable, we effectively alter the complexity of the exploration, transforming potential mission failure into a tractable, safety-first learning process. This pruning mechanism drastically reduces the explored state and action spaces, directly improving sample complexity. For instance, in the Schedule environment, our lexicographical filter efficiently restricts exploration to just 198.0 states and 320.5 transitions, achieving an approximate 82.4% reduction in explored states and an 82.3% reduction in transitions compared to the resources wasted by standard Q-learning. These findings provide a rigorous foundation for future research in goal-oriented RL, suggesting that goal-reachability exploration is essential for scaling reliable agents to high-risk domains. Future work will focus on scaling these lexicographical principles to deep reinforcement learning architectures and exploring more complex, high-dimensional state representations.

References

Achiam, J.; Held, D.; Tamar, A.; and Abbeel, P. 2017. Constrained Policy Optimization. arXiv:1705.10528.

Agarwal, R.; Schwarzer, M.; Castro, P. S.; Courville, A.; and Bellemare, M. G. 2021. Deep Reinforcement Learning at the Edge of the Statistical Precipice. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, 29304–29320. Red Hook, NY, USA: Curran Associates Inc. ISBN 978-1-7138-4539-3.

Altman, E. 1999. *Constrained Markov Decision Processes*. CRC Press. ISBN 978-0-8493-0382-1.

Auer, P.; Cesa-Bianchi, N.; and Fischer, P. 2002. Finite-Time Analysis of the Multiarmed Bandit Problem. *Machine Learning*, 47(2): 235–256.

Bagatella, M.; Krause, A.; and Martius, G. 2025. Directed Exploration in Reinforcement Learning from Linear Temporal Logic. arXiv:2408.09495.

Bellman, R. 1957. *Dynamic Programming*. Princeton University Press. ISBN 978-0-691-07951-6.

Bertsekas, D. 2022. *Abstract Dynamic Programming: 3rd Edition*. Athena Scientific. ISBN 978-1-886529-47-2.

Bertsekas, D. 2023. *A Course in Reinforcement Learning*. Athena Scientific. ISBN 978-1-886529-49-6.

Bertsekas, D. P. 2017. *Dynamic Programming and Optimal Control*. Nashua, NH: Athena Scientific, 4th edition. ISBN 978-1-886529-43-4.

Bertsekas, D. P.; and Tsitsiklis, J. N. 1991. An Analysis of Stochastic Shortest Path Problems. *Mathematics of Operations Research*, 16(3): 580–595.

Bryce, D.; and Buffet, O. 2008a. 6th International Planning Competition: Uncertainty Part. In *Proceedings of the 6th International Planning Competition (IPC'08)*.

Bryce, D. B.; and Buffet, O. 2008b. International Planning Competition Uncertainty Part: Benchmarks and Results.

Buffet, O. 2007. Reachability Analysis for Uncertain SSPs. *International Journal on Artificial Intelligence Tools*, 16(04): 725–749.

Colas, C.; Karch, T.; Sigaud, O.; and Oudeyer, P.-Y. 2022. Autotelic Agents with Intrinsically Motivated Goal-Conditioned Reinforcement Learning: A Short Survey. arXiv:2012.09830.

Engstrom, L.; Ilyas, A.; Santurkar, S.; Tsipras, D.; Janoos, F.; Rudolph, L.; and Mądry, A. 2020. Implementation Matters in Deep RL: A Case Study on PPO and TRPO. In *International Conference on Learning Representations*.

Fatemi, M.; and Sharma, S. 2019. Dead-Ends and Secure Exploration in Reinforcement Learning. In *Proceedings of the 36th International Conference on Machine Learning*, ICML '19.

Garcıa, J.; and Fernandez, F. 2015. A Comprehensive Survey on Safe Reinforcement Learning. *Journal of Machine Learning Research*, 16(1): 1437–1480.

Geffner, H.; and Bonet, B. 2013. *A Concise Introduction to Models and Methods for Automated Planning*. Morgan & Claypool Publishers. ISBN 978-1-60845-970-4.

Ghallab, M.; Nau, D.; and Traverso, P. 2004. *Automated Planning: Theory and Practice*. Elsevier. ISBN 978-1-55860-856-6.

Hayes, C. F.; Rădulescu, R.; Bargiacchi, E.; Kallstrom, J.; Macfarlane, M.; Reymond, M.; Verstraeten, T.; Zintgraf, L. M.; Dazeley, R.; Heintz, F.; Howley, E.; Irissappane, A. A.; Mannion, P.; Nowé, A.; Ramos, G.; Restelli, M.; Vamplew, P.; and Roijers, D. M. 2023. A Brief Guide to Multi-Objective Reinforcement Learning and Planning. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '23, 1988–1990. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 978-1-4503-9432-1.

Henderson, P.; Islam, R.; Bachman, P.; Pineau, J.; Precup, D.; and Meger, D. 2018. Deep Reinforcement Learning That Matters. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*.

Kaelbling, L. 1993. Learning to Achieve Goals. In *International Joint Conference on Artificial Intelligence*.

Kolobov, A. 2013. *Scalable Methods and Expressive Models for Planning Under Uncertainty*. Thesis.

Kolobov, A.; Mausam, M.; Weld, D.; and Geffner, H. 2011. Heuristic Search for Generalized Stochastic Shortest Path MDPs. *Proceedings of the International Conference on Automated Planning and Scheduling*, 21: 130–137.

Kolobov, A.; and Weld, D. S. 2012. A Theory of Goal-Oriented MDPs with Dead Ends. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, UAI'12.

Kwiatkowska, M.; Norman, G.; and Parker, D. 2022. Probabilistic Model Checking and Autonomy. *Annual Review of Control, Robotics, and Autonomous Systems*, 5(1): 385–410.

Little, I.; and Thiebaux, S. 2007. Probabilistic Planning vs Replanning.

Mausam; and Kolobov, A. 2012. *Planning with Markov Decision Processes: An AI Perspective*. Morgan & Claypool Publishers. ISBN 978-1-60845-886-8.

Moerland, T. M.; Broekens, J.; Plaat, A.; and Jonker, C. M. 2023. Model-Based Reinforcement Learning: A Survey. *Foundations and Trends® in Machine Learning*, 16(1): 1–118.

Puterman, M. L. 2014. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons. ISBN 978-1-118-62587-3.

Singh, S.; Jaakkola, T.; Littman, M. L.; and Szepesvári, C. 2000. Convergence Results for Single-Step On-Policy Reinforcement-Learning Algorithms. *Machine Learning*, 38(3): 287–308.

Steinmetz, M.; Hoffmann, J.; and Buffet, O. 2016. Goal Probability Analysis in Probabilistic Planning: Exploring and Enhancing the State of the Art. *Journal of Artificial Intelligence Research*, 57: 229–271.

Strehl, A. L.; and Littman, M. L. 2008. An Analysis of Model-Based Interval Estimation for Markov Decision Processes. *Journal of Computer and System Sciences*, 74(8): 1309–1331.

Sutton, R. S. 1988. Learning to Predict by the Methods of Temporal Differences. *Machine Learning*, 3(1): 9–44.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement Learning, Second Edition: An Introduction*. MIT Press. ISBN 978-0-262-03924-6.

Tarbouriech, J. 2022. *Goal-Oriented Exploration for Reinforcement Learning*. Ph.D. thesis.

Teichteil-Königsbuch, F. 2012. Stochastic Safest and Shortest Path Problems. *Proceedings of the AAAI Conference on Artificial Intelligence*, 26(1): 1825–1831.

Thrun, S. B. 1992. Efficient Exploration In Reinforcement Learning. Technical Report, Carnegie Mellon University, USA.

Trevizan, F.; Teichteil-Konigsbuch, F.; and Thiebaux, S. 2017. Efficient Solutions for Stochastic Shortest Path Problems with Dead Ends. In *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence*, UAI'17.

Vamplew, P.; Dazeley, R.; Berry, A.; Issabekov, R.; and Dekker, E. 2011. Empirical Evaluation Methods for Multiobjective Reinforcement Learning Algorithms. *Machine Learning*, 84(1): 51–80.

van Hasselt, H.; Doron, Y.; Strub, F.; Hessel, M.; Sonnerat, N.; and Modayil, J. 2018. Deep Reinforcement Learning and the Deadly Triad. arXiv:1812.02648.

Watkins, C. J. C. H.; and Dayan, P. 1992. Q-Learning. *Machine Learning*, 8(3): 279–292.

Whitehead, S. 1991. A Study of Cooperative Mechanisms for Faster Reinforcement Learning.

Younes, H. L. S.; and Littman, M. 2004. PPDDL 1 . 0 : An Extension to PDDL for Expressing Planning Domains with Probabilistic Effects.

Younes, H. L. S.; Littman, M. L.; Weissman, D.; and Asmuth, J. 2005. The First Probabilistic Track of the International Planning Competition. *Journal of Artificial Intelligence Research*, 24: 851–887.

Appendix

Compilation of Discounted MDPs into SSPs

As discussed in Section , the SSP framework is a strictly more expressive model than the standard Infinite-Horizon Discounted Markov Decision Process (MDP). Any discounted MDP can be exactly compiled into an equivalent SSP such that the optimal policy for the SSP is also optimal for the original MDP.

Formally, let $\mathcal{M}_\gamma = \langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma \rangle$ be an infinite-horizon discounted MDP with discount factor $\gamma \in [0, 1)$. We can construct an equivalent SSP $\mathcal{M}_{\text{SSP}} = \langle \mathcal{S} \cup \{s_g\}, \mathcal{A}, \mathcal{T}', \mathcal{C}, s_0, \mathcal{G} = \{s_g\} \rangle$ by introducing a special goal state $s_g \notin \mathcal{S}$ and defining the transition and cost functions as follows:

1. **Transition Dynamics:** For every state $s \in \mathcal{S}$, action $a \in \mathcal{A}$, and next state $s' \in \mathcal{S}$, the transition probability in the SSP is scaled by the discount factor:
$$\mathcal{T}'(s, a, s') = \gamma \mathcal{T}(s, a, s'). \tag{3}$$
Additionally, every state-action pair has a probability of transitioning directly to the goal state s_g:
$$\mathcal{T}'(s, a, s_g) = 1 - \gamma. \tag{4}$$
The goal state s_g is an absorbing terminal state: $\mathcal{T}'(s_g, a, s_g) = 1$ for all $a \in \mathcal{A}$.
2. **Cost Function:** The rewards from the original MDP are transformed into costs. For all $s, s' \in \mathcal{S}$ and $a \in \mathcal{A}$:
$$\mathcal{C}(s, a, s') = -\mathcal{R}(s, a, s'). \tag{5}$$
Transitions to the goal state s_g incur no cost: $\mathcal{C}(s, a, s_g) = 0$ for all $s \in \mathcal{S}, a \in \mathcal{A}$. Costs from the goal state are also zero: $\mathcal{C}(s_g, a, s_g) = 0$.

Under this construction, the expected cumulative cost in $\mathcal{M}_{\text{SSP}}$ starting from state s following policy π is:

$$V^\pi_{\text{SSP}}(s) = \mathbb{E}_\pi \left[\sum_{t=0}^{\zeta - 1} \mathcal{C}(s_t, a_t, s_{t+1}) \mid s_0 = s \right], \tag{6}$$

where ζ is the random variable representing the time step at which the agent transitions to the goal state s_g. Due to the geometric distribution of ζ induced by the $1 - \gamma$ transition probability, this expectation is equivalent to the discounted sum:

$$V^\pi_{\text{SSP}}(s) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t \mathcal{C}(s_t, a_t, s_{t+1}) \mid s_0 = s \right] = -V^\pi_\gamma(s). \tag{7}$$

Thus, minimizing the expected cost in the SSP $\mathcal{M}_{\text{SSP}}$ is mathematically identical to maximizing the expected discounted reward in the MDP $\mathcal{M}_\gamma$. This construction demonstrates that the effective horizon of a discounted MDP is captured by the goal-reachability dynamics of the equivalent SSP, where the agent "terminates" with probability $1 - \gamma$ at each step. This transformation is a standard result in the planning literature (Kolobov and Weld 2012; Geffner and Bonet 2013) and serves to unify discounted and undiscounted reinforcement learning under the SSP umbrella. Consequently, this mapping formally establishes that the class of infinite-horizon discounted MDPs is a subset of SSP problems.

Environments Description

We utilize several benchmarks to test specific aspects of SSPs with dead-ends, ranging from grid-world navigation to complex symbolic planning tasks.

Grid-I (Closing Doors). (Teichteil-Königsbuch 2012) consists of a 13×13 grid partitioned into four rooms. To reach the goal in the top-left quadrant from the start in the bottom-right, the agent must cross one of four central doors. Each door crossing carries a 25% risk of triggering a mechanism that permanently locks all doors. If triggered, the agent is trapped in an absorbing dead-end state where it incurs a perpetual penalty ($c = 2$ or 4 per step) until the time limit.

Grid-II (Vanishing Goal). (Teichteil-Königsbuch 2012) uses the same 13×13 layout as Grid-I, but with a different failure mechanic. Crossing the central doors does not block movement; instead, it carries a 25% risk of the goal state permanently disappearing from the environment. Once the goal vanishes, the agent enters an absorbing dead-end, representing a failure to reach the goal where the objective is no longer attainable.

Navigation-SSPADE. This environment features a 3×3 grid where the middle row acts as a high-risk region. Moving into or within this row carries a failure probability that scales linearly from 0% to 90% across the x-axis (columns). Because the top and bottom rows are fully reliable (failure probability 0%), a reliable path to the goal always exists. This benchmark tests the agent's ability to discover and stick to these reliable routes despite the availability of shorter, risk-prone alternatives through the middle row.

Navigation-SSPUDE. A variation of the 3×3 grid where the high-risk area in the middle row has a minimum failure probability of 10%, scaling up to 90% in the last column. In this configuration, no fully reliable path to the goal exists if the agent must cross or enter the middle row to reach the goal in a timely manner. This environment tests the agent's behavior under unavoidable dead-ends.

Exploding-Blocksworld. (Bryce and Buffet 2008b; Younes and Littman 2004) is a symbolic Probabilistic PDDL domain requiring the agent to stack blocks into a target configuration. Actions are highly risk-prone: placing a block on the table carries a 40% risk of destroying both the block and the table, while stacking a block on another has a 10% probability of destroying the underlying block. Once a block necessary for the goal is destroyed or the table becomes unusable, the goal becomes unreachable, creating a dead-end.

Search-and-Rescue. (Bryce and Buffet 2008b) simulates an autonomous helicopter mission to rescue a survivor. The agent must locate potential landing zones, *explore* them to determine landability (70% success), and perform a *land* action. Boarding the survivor carries a notable risk (20% probability of the survivor dying), and subsequent transport back to the control center (*base*) involves further risk during transit. The mission fails permanently if the survivor is lost, requiring the agent to carefully navigate the trade-off between path efficiency and mission safety across a graph-based map.

Triangle-Tireworld. (Little and Thiebaux 2007) involves

navigating a car through a network of directional roads to reach a destination. Each move carries a 50% risk of a flat tire. If the car gets a flat tire and has no spare, it becomes permanently stranded in a dead-end. The agent can mitigate this risk by carrying a single spare tire and visiting locations with stockpiles to replenish it. Finding the best policy requires balancing path length against the distribution of spares and the current inventory, as shorter routes are often more dangerous.

Schedule. (Bryce and Buffet 2008b) is a simulation of a packet-scheduling problem where the agent must process specific goal types of packets. While some types are delivered rarely, the overall arrival rate often exceeds the processing capacity. When the queue overflows, the agent must engage in high-risk recovery actions that carry a notable probability of failure (simulated as the agent's "death"). Success requires the agent to prioritize hard-to-come-by packets immediately upon delivery to prevent queue saturation and avoid dead-ends.

River. (Little and Thiebaux 2007) tasks an agent with crossing from one side of a river to the other. The agent can attempt to traverse slippery rocks, which carry a 75% chance of slipping; a fall results in drowning (death) with probability $1/3$ or reaching a small island in the middle of the river with probability $2/3$. Alternatively, the agent can swim across from the near bank with an even (50%) chance of success. If the agent reaches the island, it has a higher success rate (80%) of swimming to the far bank but faces a 20% risk of drowning. This benchmark evaluates the agent's ability to maximize its chance of reaching the goal without drowning, as the optimal policy must balance immediate progress against long-term survival risks.

Experimental Hyperparameters

The following table summarizes the hyperparameters for the exploration benchmark evaluations.

Table 4: Hyperparameters for Exploration Benchmark

Category	Hyperparameter	Value(s)
Common	Number of Seeds	30
	Evaluation Episodes	100
	Parallel Workers	12
Dual Dyna-Q	Learning Rate (α)	0.1
	Discount Factor (γ)	0.99
	Planning Steps	50
	Exploration Strategy	ϵ-greedy
	Exploration (ϵ)	1
	Exploration Decay	0.999
	Minimum Exploration	0.1
Dyna-Q	Learning Rate (α)	0.1
	Discount Factor (γ)	1
	Planning Steps	10
	Exploration Strategy	ϵ-greedy, Boltzmann, MBIE-EB, UCB
	Exploration (ϵ)	1
	Exploration Decay	0.999
	Minimum Exploration	0.1
	Exploration Weight (β)	1, 5
FP Q-learning	Learning Rate (α)	0.1
	Discount Factor (γ)	0.99
	Finite Penalty (D)	100
	Exploration Strategy	ϵ-greedy, Boltzmann, UCB
	Exploration (ϵ)	1
	Exploration Decay	0.999
	Minimum Exploration	0.1
	Exploration Weight (β)	1, 5
Q-learning	Learning Rate (α)	0.1
	Discount Factor (γ)	0.99
	Exploration Strategy	ϵ-greedy, Boltzmann, UCB
	Exploration (ϵ)	1
	Exploration Decay	0.999
	Minimum Exploration	0.1
	Exploration Weight (β)	1, 5

Detailed Experimental Results

In this section, we present the comprehensive performance metrics for all evaluated algorithms across the benchmark environments.

Table 5: Mean episode length at the end of training for all benchmark environments.

Algorithm	Exploration	Environment								
		Exploding-Blocksworld	Grid-I	Grid-II	Navigation-SSPADE	Navigation-SSPUDE	River	Schedule	Search-and-Rescue	Triangle-Tireworld
Dual Dyna-Q	lex ϵ-greedy	28.706 ± 24.228	$\mathbf{101.072 \pm 20.613}$	$\mathbf{118.027 \pm 38.833}$	$\mathbf{6.000 \pm 0.000}$	31.220 ± 20.737	1.343 ± 0.234	30.186 ± 2.882	35.891 ± 23.704	6.351 ± 0.243
Dyna-Q	ϵ-greedy	$\mathbf{14.903 \pm 6.341}$	200.000 ± 0.000	200.000 ± 0.000	$\mathbf{6.000 \pm 0.000}$	188.166 ± 45.042	$\mathbf{1.000 \pm 0.000}$	29.089 ± 2.663	18.571 ± 12.651	1.494 ± 0.056
	Boltzmann	141.821 ± 44.757	200.000 ± 0.000	200.000 ± 0.000	$\mathbf{6.000 \pm 0.000}$	194.051 ± 32.586	$\mathbf{1.000 \pm 0.000}$	30.008 ± 2.305	$\mathbf{10.623 \pm 4.244}$	$\mathbf{1.490 \pm 0.056}$
	MBIE-EB	16.387 ± 6.949	200.000 ± 0.000	200.000 ± 0.000	$\mathbf{6.000 \pm 0.000}$	200.000 ± 0.000	$\mathbf{1.000 \pm 0.000}$	28.966 ± 2.336	13.924 ± 5.274	1.516 ± 0.050
	UCB	26.919 ± 22.357	200.000 ± 0.000	200.000 ± 0.000	$\mathbf{6.000 \pm 0.000}$	188.554 ± 43.566	$\mathbf{1.000 \pm 0.000}$	30.622 ± 2.880	11.214 ± 4.455	1.513 ± 0.047
FP Q-learning	ϵ-greedy	163.450 ± 40.312	200.000 ± 0.000	200.000 ± 0.000	$\mathbf{6.000 \pm 0.000}$	113.282 ± 88.262	1.497 ± 0.058	30.066 ± 1.920	158.614 ± 40.071	6.195 ± 0.208
	Boltzmann	171.752 ± 35.119	200.000 ± 0.000	200.000 ± 0.000	$\mathbf{6.000 \pm 0.000}$	29.927 ± 32.520	1.493 ± 0.057	$\mathbf{28.884 \pm 3.030}$	145.817 ± 38.797	6.228 ± 0.209
	UCB	143.133 ± 41.491	200.000 ± 0.000	200.000 ± 0.000	$\mathbf{6.000 \pm 0.000}$	26.111 ± 6.454	1.493 ± 0.057	30.607 ± 2.552	120.673 ± 24.441	6.169 ± 0.242
Q-learning	ϵ-greedy	184.656 ± 31.318	200.000 ± 0.000	200.000 ± 0.000	$\mathbf{6.000 \pm 0.000}$	153.505 ± 78.447	$\mathbf{1.000 \pm 0.000}$	31.371 ± 2.693	55.664 ± 15.452	1.508 ± 0.063
	Boltzmann	152.623 ± 41.116	200.000 ± 0.000	200.000 ± 0.000	$\mathbf{6.000 \pm 0.000}$	29.603 ± 32.793	$\mathbf{1.000 \pm 0.000}$	$\mathbf{28.884 \pm 3.030}$	44.619 ± 17.770	1.502 ± 0.053
	UCB	147.398 ± 41.058	200.000 ± 0.000	200.000 ± 0.000	$\mathbf{6.000 \pm 0.000}$	$\mathbf{24.301 \pm 4.957}$	$\mathbf{1.000 \pm 0.000}$	30.607 ± 2.552	20.410 ± 10.876	1.493 ± 0.057

Table 6: Total unique states explored during training (mean $\pm$ standard deviation across 30 independent seeds).

Algorithm	Exploration	Environment								
		Exploding-Blocksworld	Grid-I	Grid-II	Navigation-SSPADE	Navigation-SSPUDE	River	Schedule	Search-and-Rescue	Triangle-Tireworld
Dual Dyna-Q	lex ϵ-greedy	11406.033 ± 290.003	87.100 ± 19.907	91.867 ± 14.438	$\mathbf{10.000 \pm 0.000}$	$\mathbf{10.000 \pm 0.000}$	4.800 ± 0.407	198.000 ± 29.253	1291.633 ± 23.841	44.100 ± 4.366
Dyna-Q	ϵ-greedy	14016.300 ± 247.736	91.333 ± 8.559	91.333 ± 8.559	$\mathbf{10.000 \pm 0.000}$	$\mathbf{10.000 \pm 0.000}$	$\mathbf{5.000 \pm 0.000}$	730.600 ± 44.329	1151.567 ± 17.250	50.567 ± 0.679
	Boltzmann	$\mathbf{14859.967 \pm 161.753}$	64.267 ± 18.844	64.267 ± 18.844	$\mathbf{10.000 \pm 0.000}$	$\mathbf{10.000 \pm 0.000}$	$\mathbf{5.000 \pm 0.000}$	817.200 ± 50.949	1337.667 ± 14.700	50.967 ± 0.183
	MBIE-EB	14351.533 ± 219.374	62.200 ± 18.427	62.200 ± 18.427	$\mathbf{10.000 \pm 0.000}$	$\mathbf{10.000 \pm 0.000}$	$\mathbf{5.000 \pm 0.000}$	405.500 ± 48.271	1169.267 ± 16.045	25.400 ± 4.415
	UCB	14798.533 ± 218.550	69.400 ± 22.263	69.400 ± 22.263	$\mathbf{10.000 \pm 0.000}$	$\mathbf{10.000 \pm 0.000}$	$\mathbf{5.000 \pm 0.000}$	897.667 ± 46.394	1328.400 ± 12.227	46.033 ± 1.542
FP Q-learning	ϵ-greedy	12822.300 ± 163.895	$\mathbf{95.900 \pm 6.150}$	$\mathbf{95.900 \pm 6.150}$	$\mathbf{10.000 \pm 0.000}$	$\mathbf{10.000 \pm 0.000}$	$\mathbf{5.000 \pm 0.000}$	775.967 ± 41.123	1436.133 ± 7.060	$\mathbf{51.000 \pm 0.000}$
	Boltzmann	8320.900 ± 173.433	89.133 ± 5.661	89.133 ± 5.661	$\mathbf{10.000 \pm 0.000}$	$\mathbf{10.000 \pm 0.000}$	$\mathbf{5.000 \pm 0.000}$	$\mathbf{1379.567 \pm 59.830}$	1476.933 ± 4.863	$\mathbf{51.000 \pm 0.000}$
	UCB	10600.267 ± 149.782	90.867 ± 6.301	90.867 ± 6.301	$\mathbf{10.000 \pm 0.000}$	$\mathbf{10.000 \pm 0.000}$	$\mathbf{5.000 \pm 0.000}$	1107.567 ± 32.707	$\mathbf{1480.233 \pm 5.144}$	50.833 ± 0.379
Q-learning	ϵ-greedy	8930.033 ± 182.351	84.600 ± 5.661	84.600 ± 5.661	$\mathbf{10.000 \pm 0.000}$	$\mathbf{10.000 \pm 0.000}$	$\mathbf{5.000 \pm 0.000}$	1123.667 ± 34.718	1454.700 ± 7.953	$\mathbf{51.000 \pm 0.000}$
	Boltzmann	11603.100 ± 151.217	88.933 ± 5.705	88.933 ± 5.705	$\mathbf{10.000 \pm 0.000}$	$\mathbf{10.000 \pm 0.000}$	$\mathbf{5.000 \pm 0.000}$	$\mathbf{1379.567 \pm 59.830}$	1469.833 ± 7.067	$\mathbf{51.000 \pm 0.000}$
	UCB	13923.500 ± 165.074	91.300 ± 6.226	91.300 ± 6.226	$\mathbf{10.000 \pm 0.000}$	$\mathbf{10.000 \pm 0.000}$	$\mathbf{5.000 \pm 0.000}$	1107.567 ± 32.707	1458.567 ± 7.555	49.233 ± 1.165

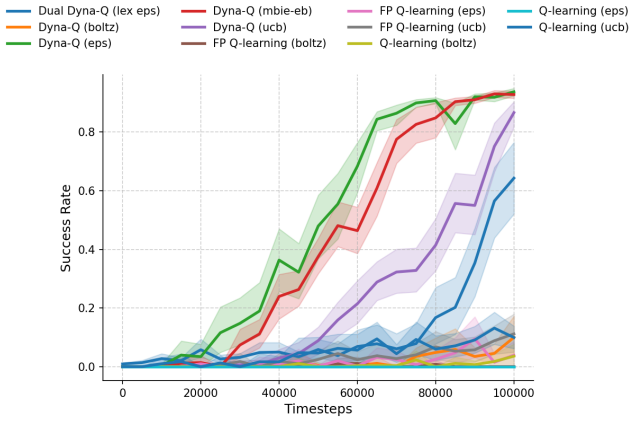
Table 7: Total unique state-action pairs (transitions) explored during training (mean $\pm$ standard deviation across 30 independent seeds).

Algorithm	Exploration	Environment								
		Exploding-Blocksworld	Grid-I	Grid-II	Navigation-SSPADE	Navigation-SSPUDE	River	Schedule	Search-and-Rescue	Triangle-Tireworld
Dual Dyna-Q	lex ϵ-greedy	17070.500 ± 439.550	428.933 ± 99.942	$\mathbf{452.600 \pm 71.138}$	$\mathbf{36.000 \pm 0.000}$	$\mathbf{36.000 \pm 0.000}$	2.800 ± 0.407	320.533 ± 44.179	3651.267 ± 75.195	63.833 ± 7.149
Dyna-Q	ϵ-greedy	22011.800 ± 371.546	433.633 ± 36.093	433.633 ± 36.093	$\mathbf{36.000 \pm 0.000}$	$\mathbf{36.000 \pm 0.000}$	$\mathbf{3.000 \pm 0.000}$	1371.333 ± 88.545	3542.933 ± 46.515	74.967 ± 2.470
	Boltzmann	22217.900 ± 214.245	303.700 ± 92.287	303.700 ± 92.287	$\mathbf{36.000 \pm 0.000}$	$\mathbf{36.000 \pm 0.000}$	$\mathbf{3.000 \pm 0.000}$	1346.667 ± 86.789	3814.233 ± 16.060	81.733 ± 0.640
	MBIE-EB	23304.367 ± 366.687	289.333 ± 84.818	289.333 ± 84.818	$\mathbf{36.000 \pm 0.000}$	$\mathbf{36.000 \pm 0.000}$	$\mathbf{3.000 \pm 0.000}$	735.833 ± 90.058	3609.233 ± 34.000	28.567 ± 5.500
	UCB	$\mathbf{24205.333 \pm 363.887}$	329.467 ± 103.130	329.467 ± 103.130	$\mathbf{36.000 \pm 0.000}$	$\mathbf{36.000 \pm 0.000}$	$\mathbf{3.000 \pm 0.000}$	1638.400 ± 84.263	3829.400 ± 9.379	60.067 ± 2.638
FP Q-learning	ϵ-greedy	20478.000 ± 254.404	$\mathbf{446.833 \pm 28.874}$	446.833 ± 28.874	$\mathbf{36.000 \pm 0.000}$	$\mathbf{36.000 \pm 0.000}$	$\mathbf{3.000 \pm 0.000}$	1401.067 ± 71.399	3928.200 ± 7.078	80.933 ± 0.365
	Boltzmann	11904.600 ± 232.060	419.667 ± 16.485	419.667 ± 16.485	$\mathbf{36.000 \pm 0.000}$	$\mathbf{36.000 \pm 0.000}$	$\mathbf{3.000 \pm 0.000}$	$\mathbf{2275.067 \pm 77.085}$	3966.667 ± 5.320	81.633 ± 0.490
	UCB	17054.200 ± 247.937	428.033 ± 23.560	428.033 ± 23.560	$\mathbf{36.000 \pm 0.000}$	$\mathbf{36.000 \pm 0.000}$	$\mathbf{3.000 \pm 0.000}$	2020.600 ± 56.215	$\mathbf{3972.267 \pm 5.206}$	81.367 ± 0.765
Q-learning	ϵ-greedy	12728.633 ± 218.308	406.733 ± 13.478	406.733 ± 13.478	$\mathbf{36.000 \pm 0.000}$	$\mathbf{36.000 \pm 0.000}$	$\mathbf{3.000 \pm 0.000}$	1808.967 ± 40.911	3938.733 ± 8.124	$\mathbf{82.000 \pm 0.000}$
	Boltzmann	16935.467 ± 191.412	418.500 ± 15.841	418.500 ± 15.841	$\mathbf{36.000 \pm 0.000}$	$\mathbf{36.000 \pm 0.000}$	$\mathbf{3.000 \pm 0.000}$	$\mathbf{2275.067 \pm 77.085}$	3960.400 ± 7.123	$\mathbf{82.000 \pm 0.000}$
	UCB	22730.800 ± 228.470	429.267 ± 21.904	429.267 ± 21.904	$\mathbf{36.000 \pm 0.000}$	$\mathbf{36.000 \pm 0.000}$	$\mathbf{3.000 \pm 0.000}$	2020.600 ± 56.215	3951.900 ± 7.146	69.233 ± 2.609

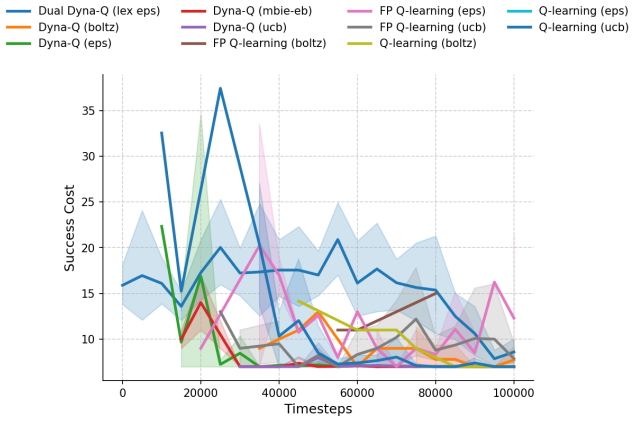
Learning Curves

In this section, we provide the complete set of learning curves for all benchmark environments. Each figure displays the mean performance across 30 independent seeds, with the shaded region representing the 95% confidence interval.

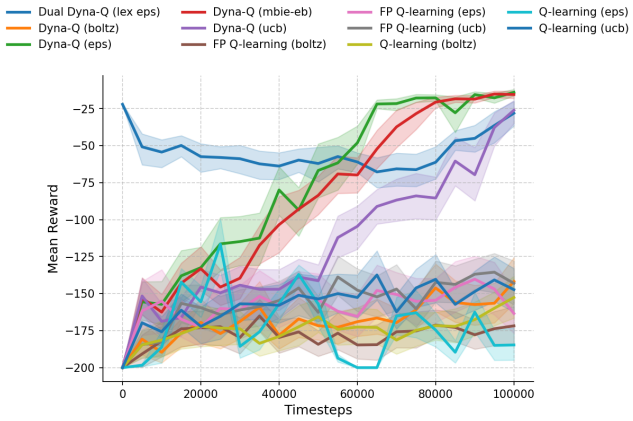
Exploding-Blocksworld



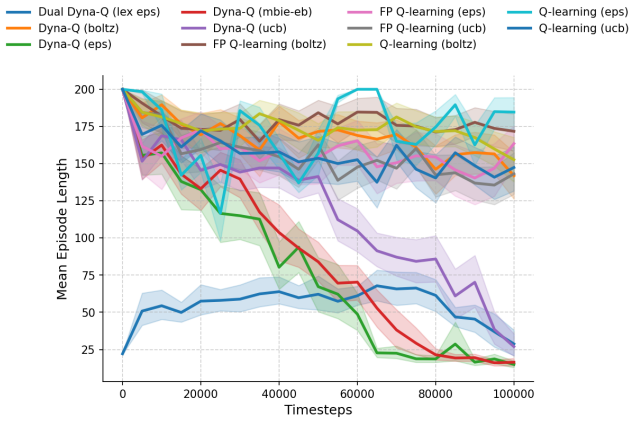
(a) Success Rate



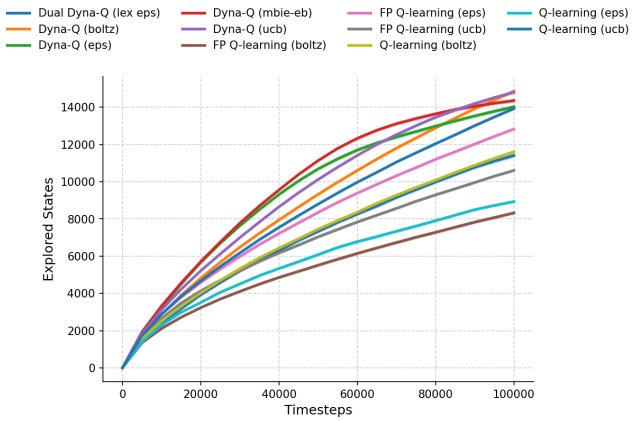
(b) Success Cost



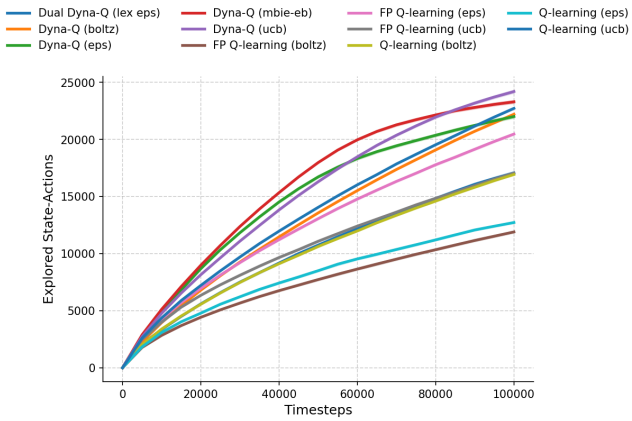
(c) Mean Reward



(d) Mean Episode Length



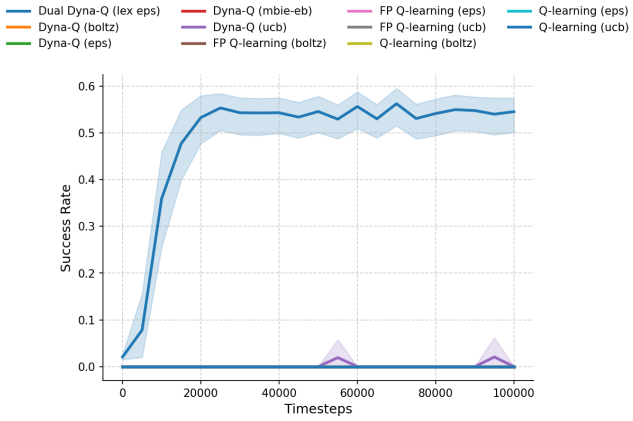
(e) Explored States



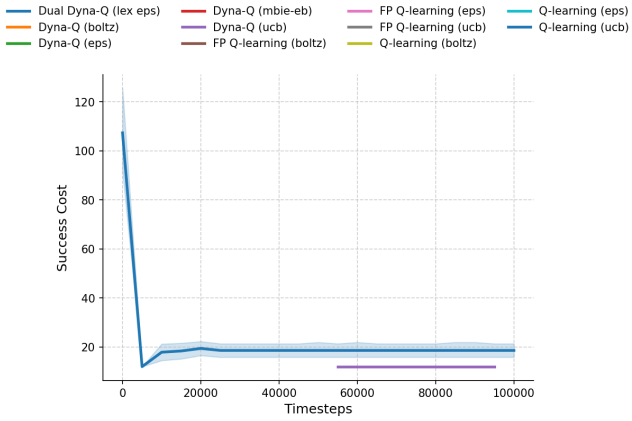
(f) Explored State-Actions

Figure 4: Learning curves for Exploding-Blocksworld across all evaluated metrics.

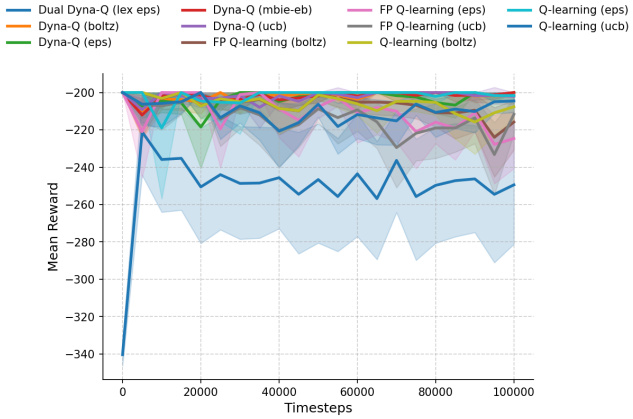
Grid-I



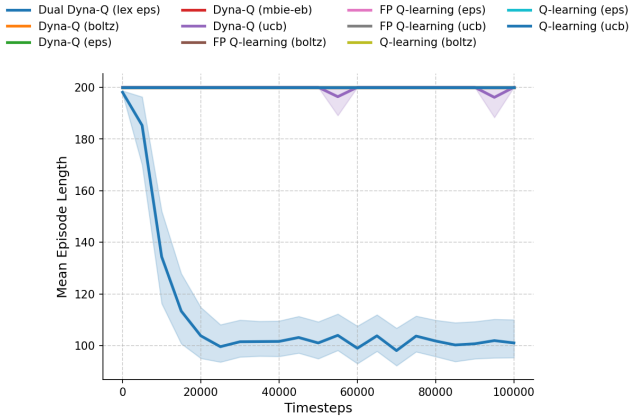
(a) Success Rate



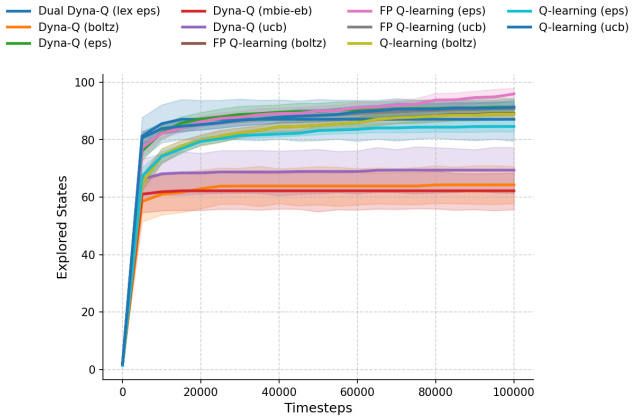
(b) Success Cost



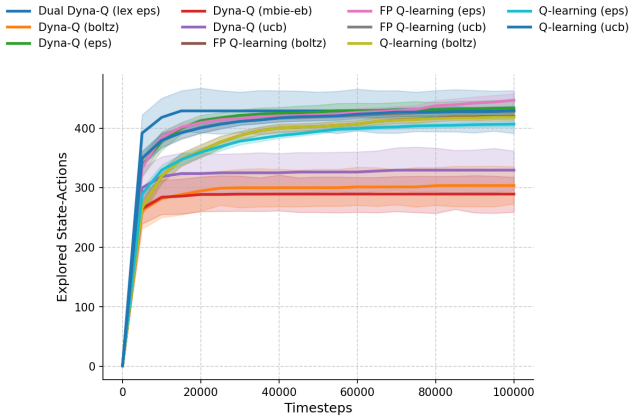
(c) Mean Reward



(d) Mean Episode Length



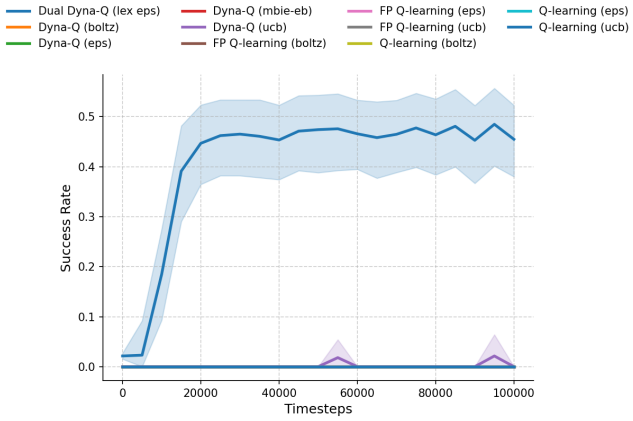
(e) Explored States



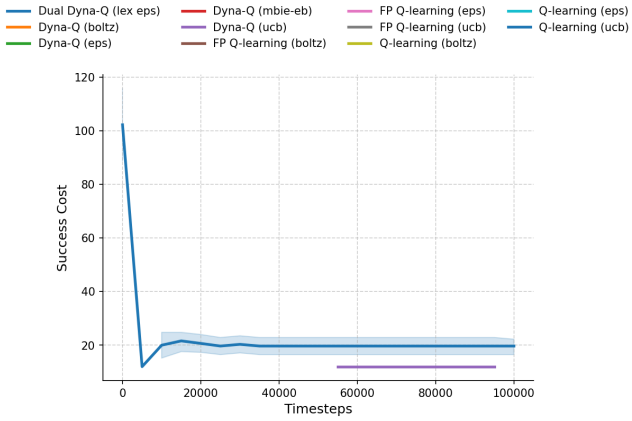
(f) Explored State-Actions

Figure 5: Learning curves for Grid-I across all evaluated metrics.

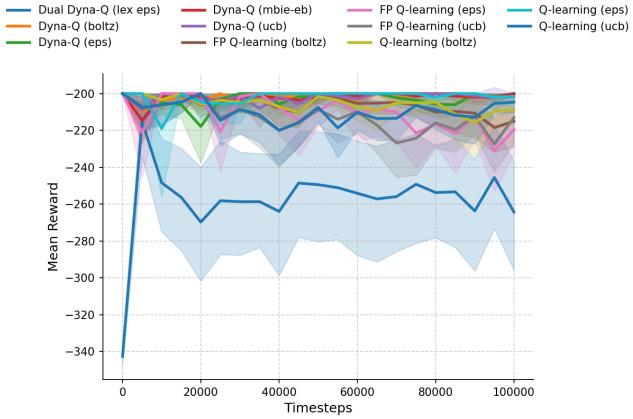
Grid-II



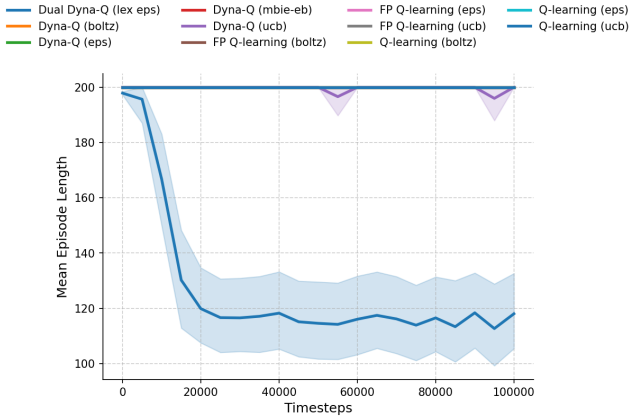
(a) Success Rate



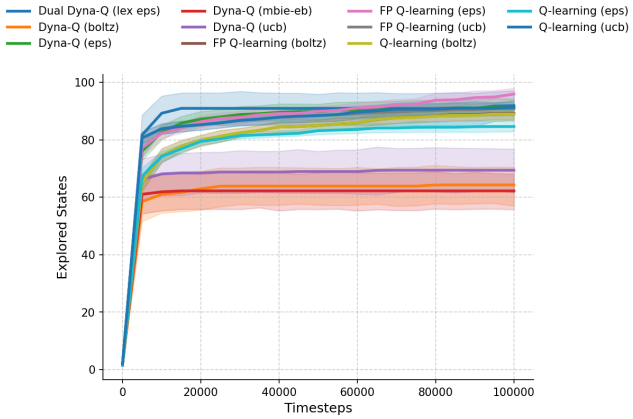
(b) Success Cost



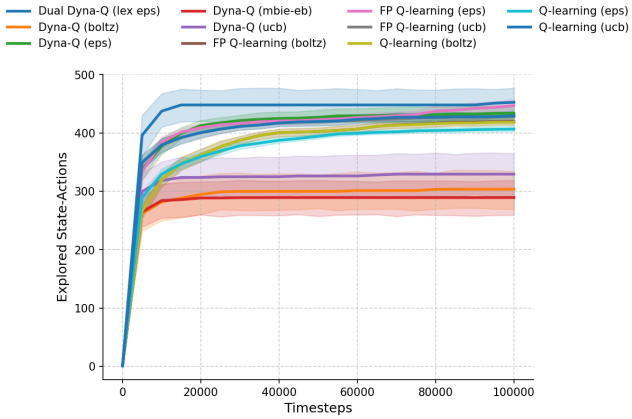
(c) Mean Reward



(d) Mean Episode Length



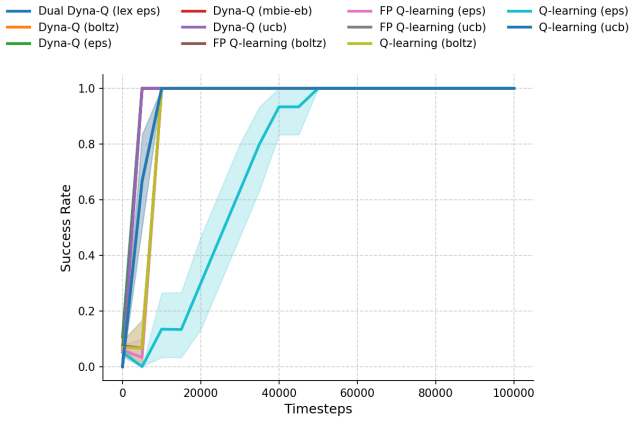
(e) Explored States



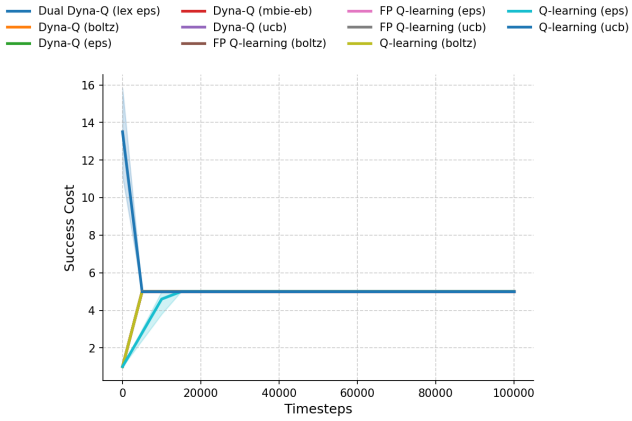
(f) Explored State-Actions

Figure 6: Learning curves for Grid-II across all evaluated metrics.

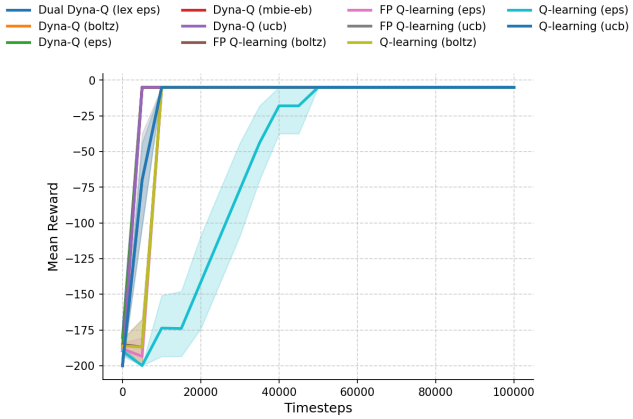
Navigation-SSPADE



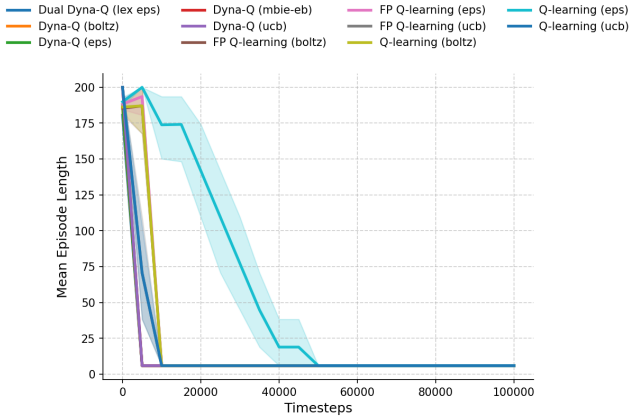
(a) Success Rate



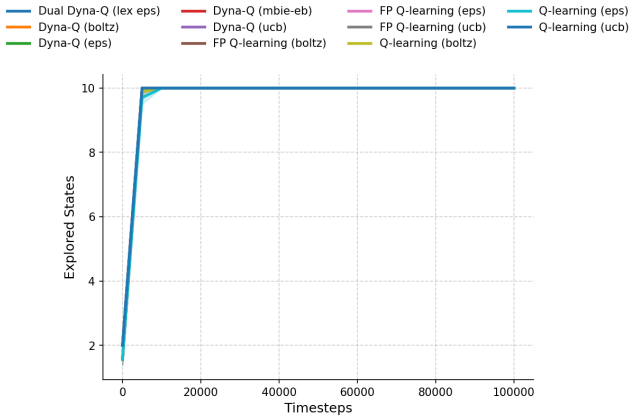
(b) Success Cost



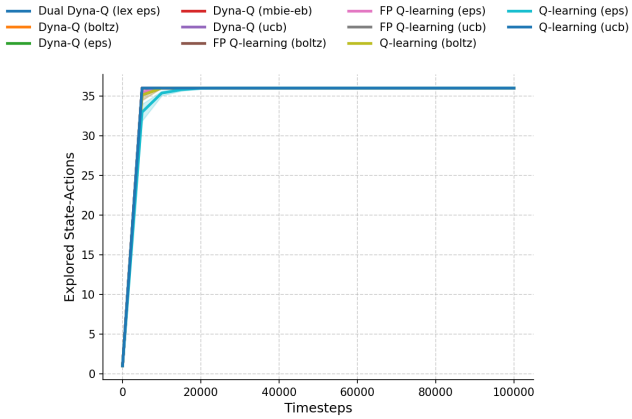
(c) Mean Reward



(d) Mean Episode Length



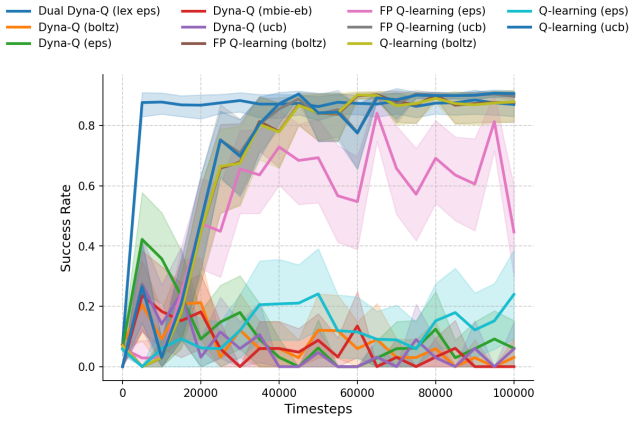
(e) Explored States



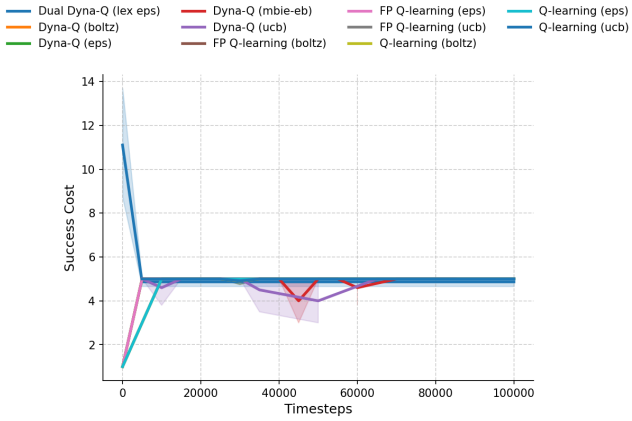
(f) Explored State-Actions

Figure 7: Learning curves for Navigation-SSPADE across all evaluated metrics.

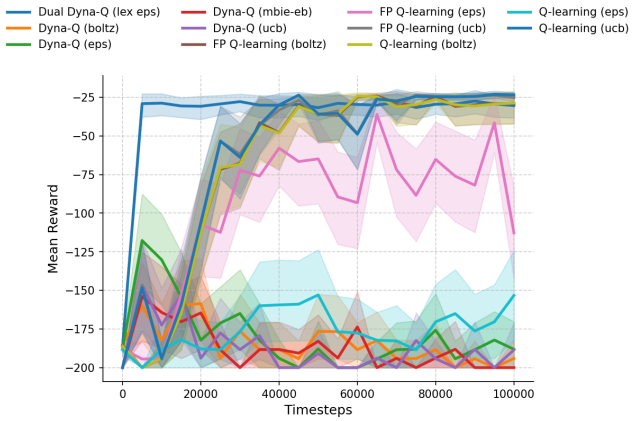
Navigation-SSPUDE



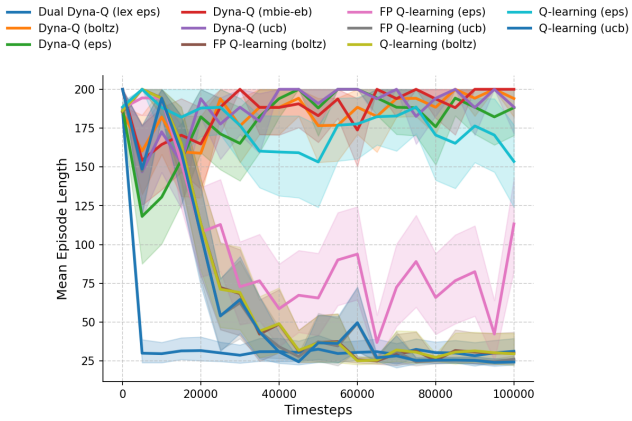
(a) Success Rate



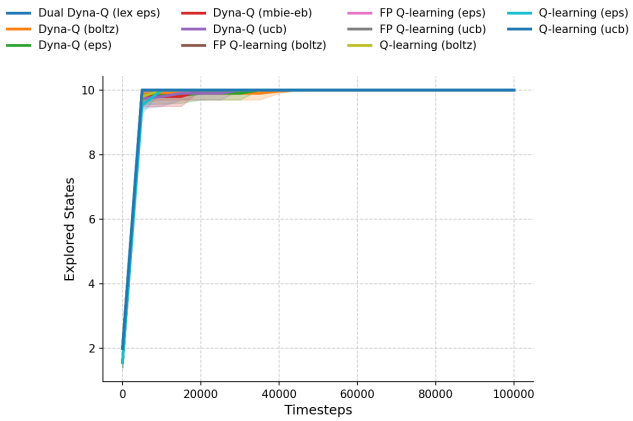
(b) Success Cost



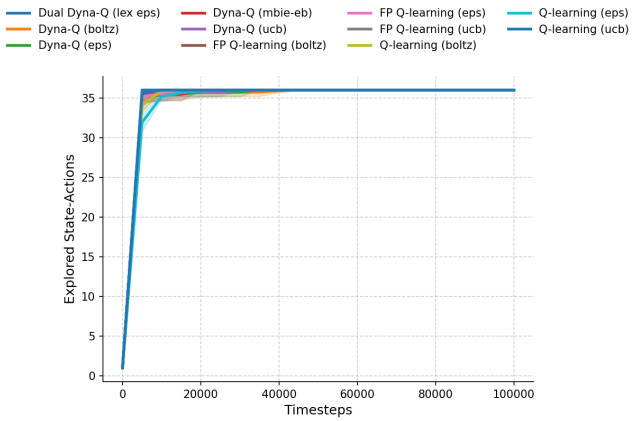
(c) Mean Reward



(d) Mean Episode Length



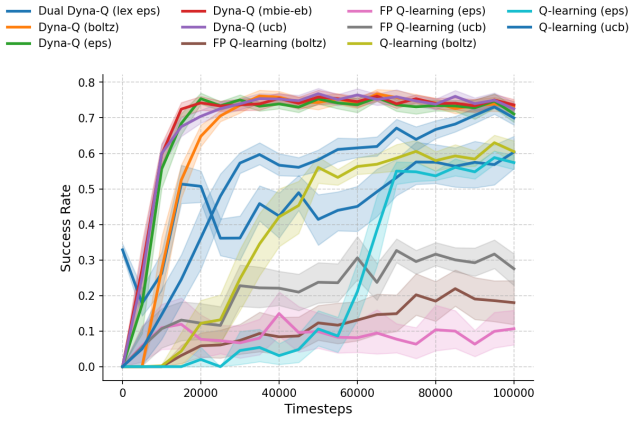
(e) Explored States



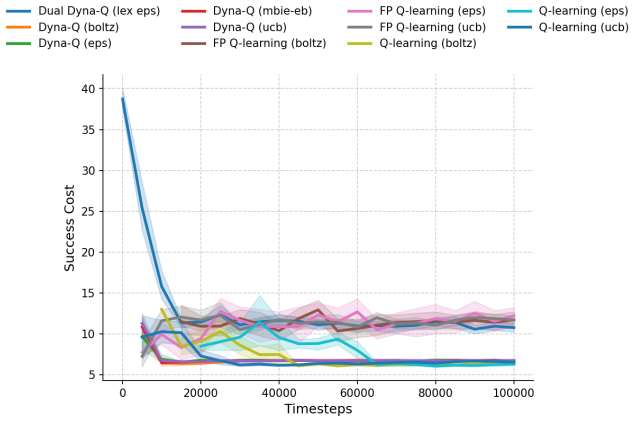
(f) Explored State-Actions

Figure 8: Learning curves for Navigation-SSPUDE across all evaluated metrics.

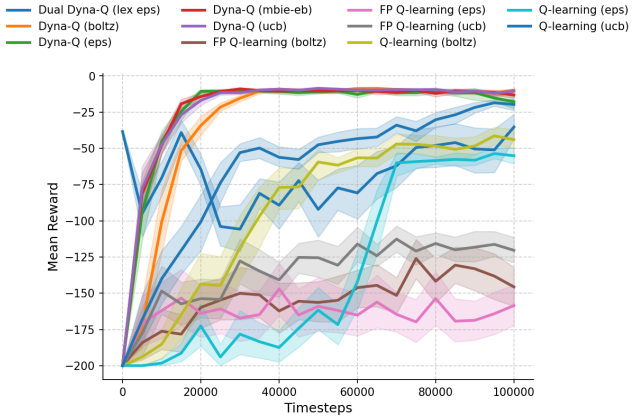
Search-and-Rescue



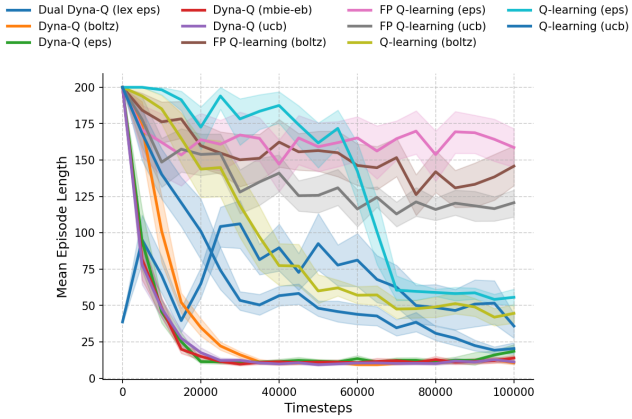
(a) Success Rate



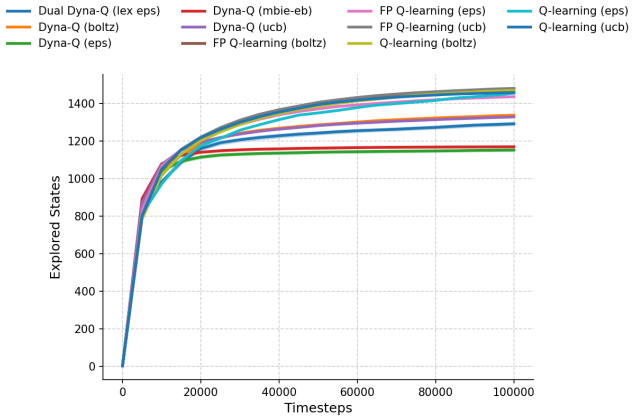
(b) Success Cost



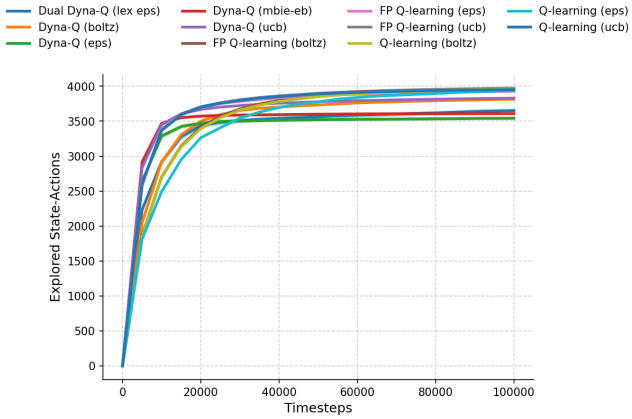
(c) Mean Reward



(d) Mean Episode Length



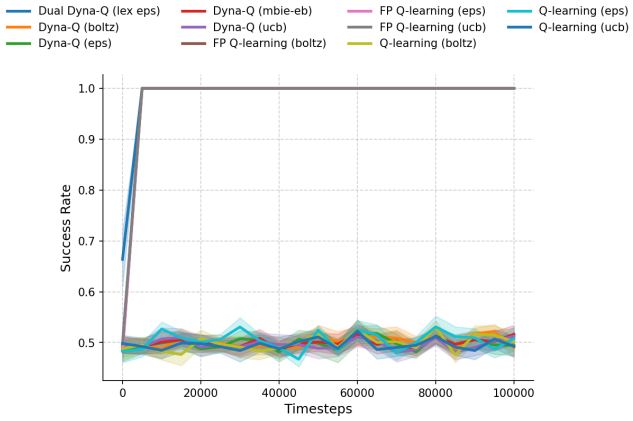
(e) Explored States



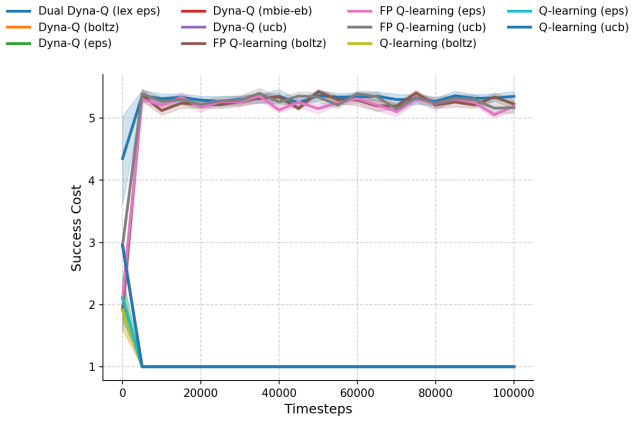
(f) Explored State-Actions

Figure 9: Learning curves for Search-and-Rescue across all evaluated metrics.

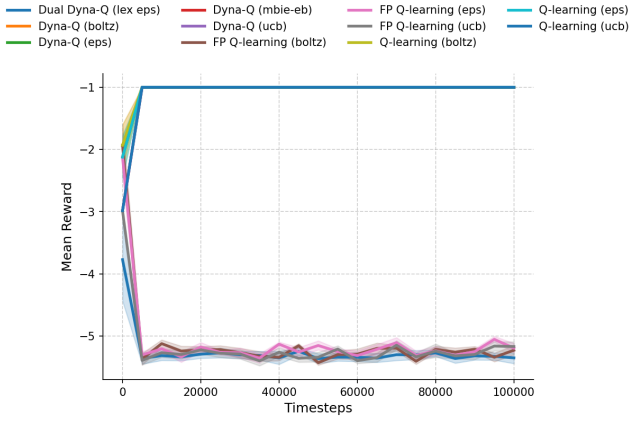
Triangle-Tireworld



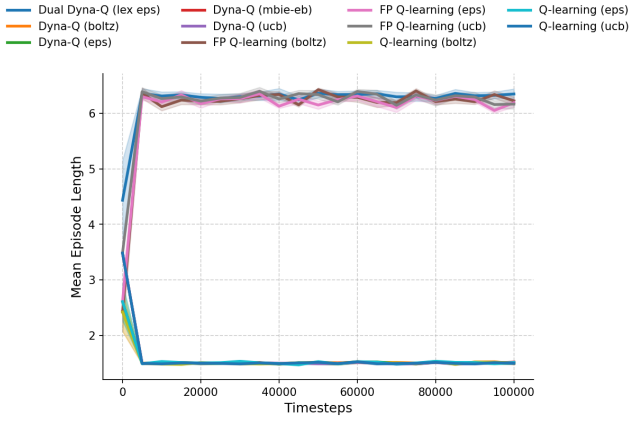
(a) Success Rate



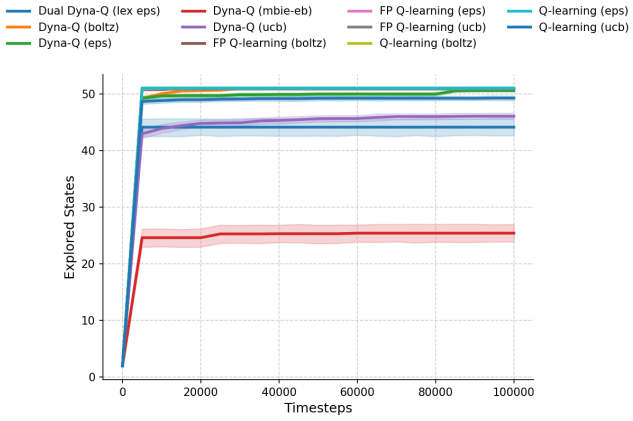
(b) Success Cost



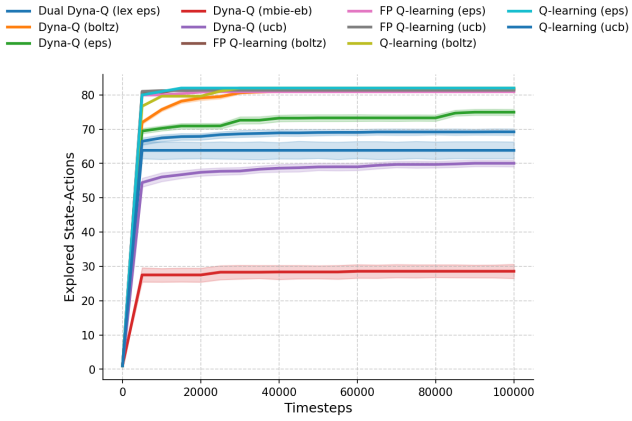
(c) Mean Reward



(d) Mean Episode Length



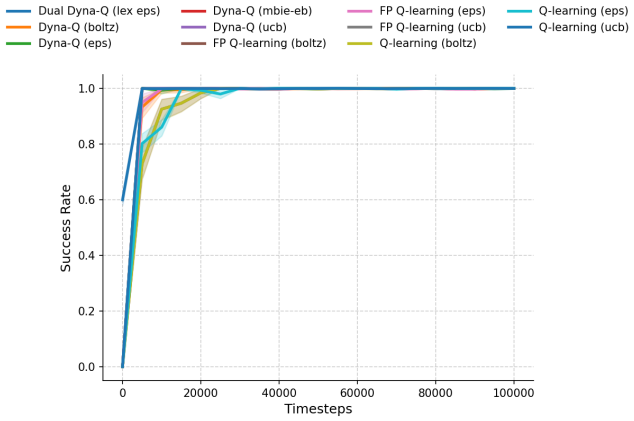
(e) Explored States



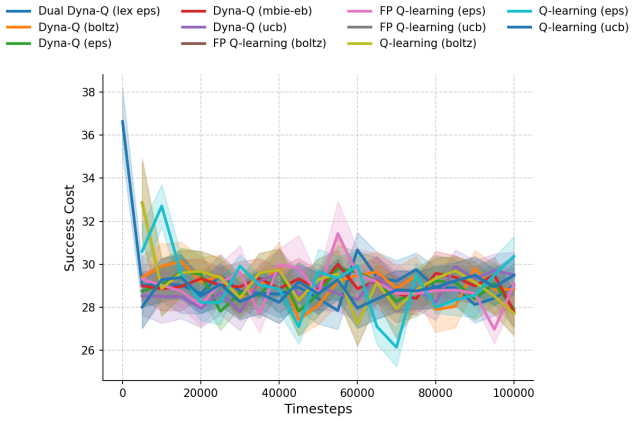
(f) Explored State-Actions

Figure 10: Learning curves for Triangle-Tireworld across all evaluated metrics.

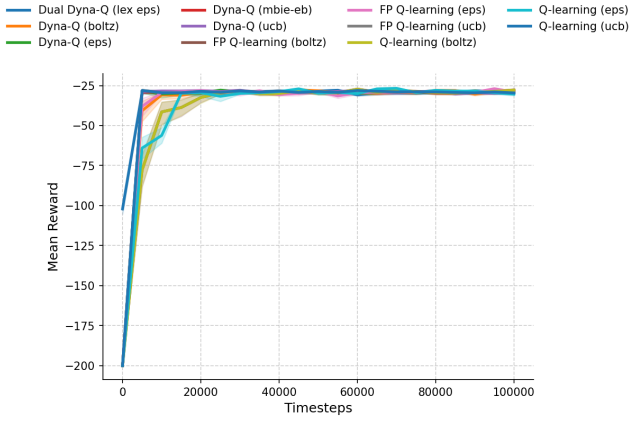
Schedule



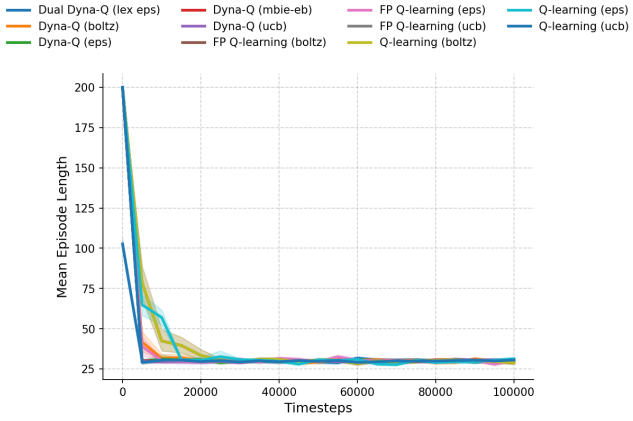
(a) Success Rate



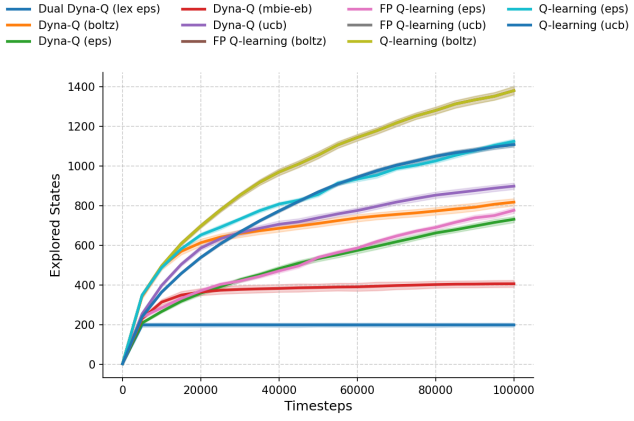
(b) Success Cost



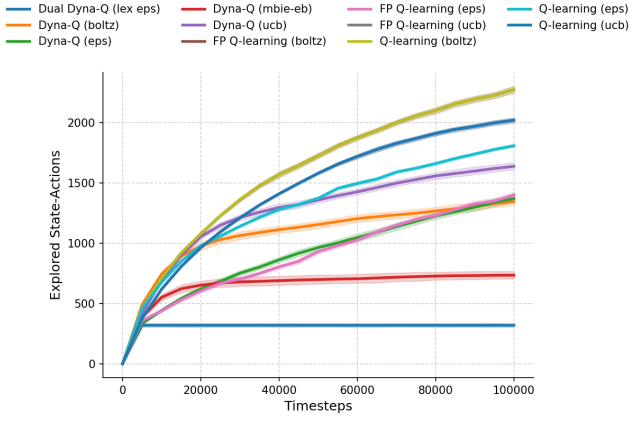
(c) Mean Reward



(d) Mean Episode Length



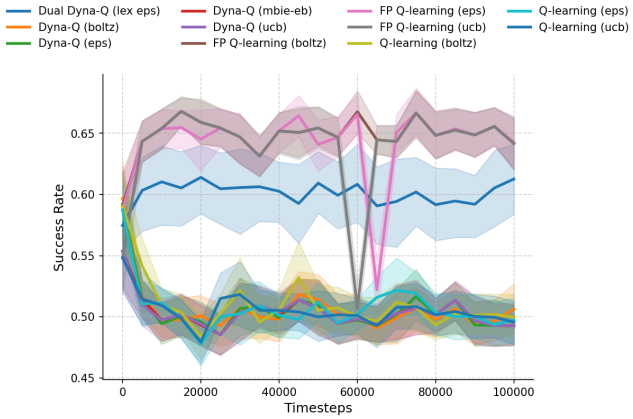
(e) Explored States



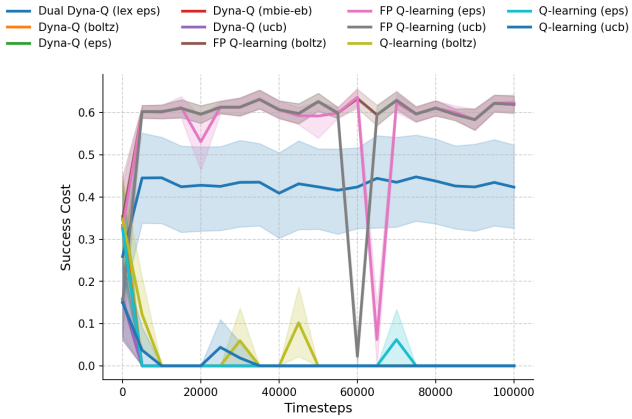
(f) Explored State-Actions

Figure 11: Learning curves for Schedule across all evaluated metrics.

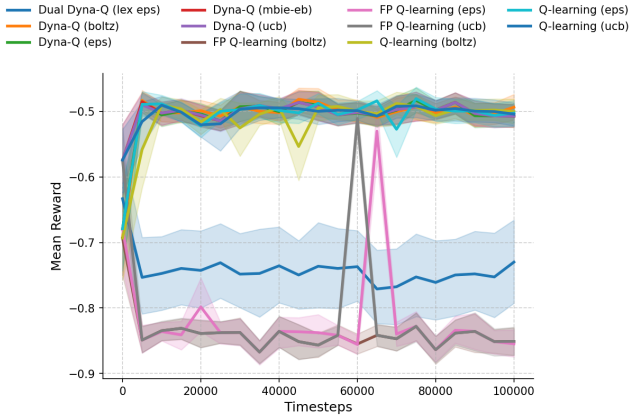
River



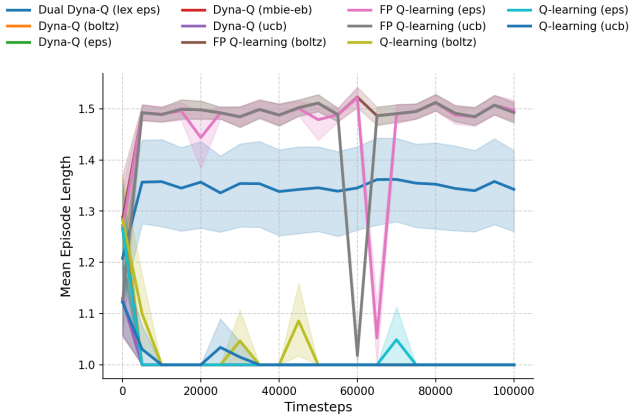
(a) Success Rate



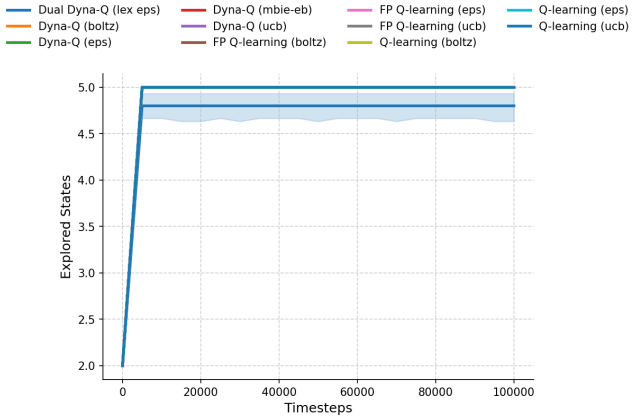
(b) Success Cost



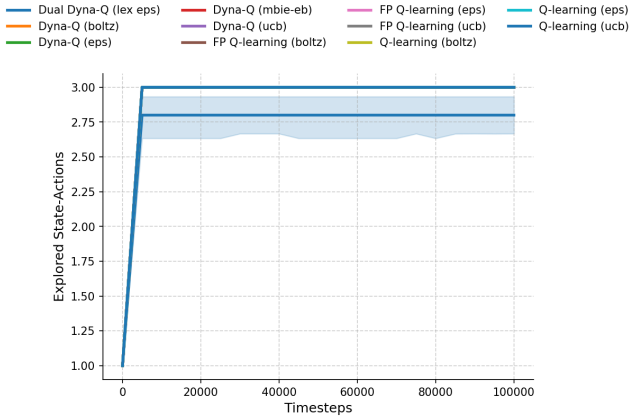
(c) Mean Reward



(d) Mean Episode Length



(e) Explored States



(f) Explored State-Actions

Figure 12: Learning curves for River across all evaluated metrics.