The Effective Horizon Challenge

Cassidy Laidlaw[*]
UC Berkeley

Daniel P. Khalil
Caltech

Michelle Li
UC Berkeley

Laker Newhouse
UC Berkeley & MIT

Stuart Russell
UC Berkeley

Anca Dragan
UC Berkeley

Abstract

While benchmarks have driven significant progress in deep reinforcement learning (RL), they may be easier to solve than intended: recent work has found that many RL benchmark environments have a short *effective horizon*, a measure of complexity that captures how easy it is to explore and solve an environment via Monte Carlo lookahead search. We introduce a new benchmark, the Effective Horizon Challenge (EHC), which consists of environments with much longer effective horizons than those in past benchmarks. Although environments in the EHC have small state spaces, short episodes, shaped rewards, and deterministic transitions, we find that deep RL struggles to solve them. For example, PPO finds an optimal policy in only 8 of 43 environments in the EHC and DQN in only 12. Our results establish environments with long effective horizons as a new frontier for deep RL research, and the Effective Horizon Challenge provides a concrete way to make progress in this direction.

1 Introduction

A decade of progress in deep reinforcement learning (RL) has been enabled by benchmarks like the Arcade Learning Environment (ALE) [1]. Deep RL algorithms have progressed from matching human performance using huge amounts of experience to surpassing human performance with minimal data [2, 3, 4, 5, 6]. However, prior work by Laidlaw et al. [7, 8] has found that the ALE and other popular deep RL benchmarks may be easier to solve than intended. In particular, they show that many environments in these benchmarks can be solved by a single step of policy iteration on the random policy. This means that it is possible to select *optimal* actions purely by approximating the Q-function of the policy that takes uniformly *random* actions. Laidlaw et al. [7] generalize this property to introduce a complexity measure called the *effective horizon*, which they find is quite low for many deep RL benchmarks. Essentially, environments with a short effective horizon do not require extensive exploration, since finding an optimal policy only requires inspecting random rollouts rather than exhaustively exploring the environment.

Since existing benchmarks have low effective horizon, it remains an open question how well deep RL performs in environments with longer effective horizons where more exploration is needed. To address this gap in RL evaluations, we introduce a new benchmark called the Effective Horizon Challenge (EHC). The EHC consists of 43 Markov decision processes (MDPs) based on classical planning problems and games. We construct complete tabular versions of all MDPs in the EHC and verify that their effective horizons are much longer than those in the ALE and Procgen [9].

We use the EHC to benchmark two of the most popular model-free deep RL algorithms: PPO [10] and DQN [2]. We measure whether each algorithm can find an optimal policy in ten million timesteps. We find that PPO succeeds in only 8 of 43 environments and DQN succeeds in only 12 of 43.

[*]Correspondance to: cassidy_laidlaw@cs.berkeley.edu

First Exploration in AI Today Workshop at ICML (EXAIT at ICML 2025).

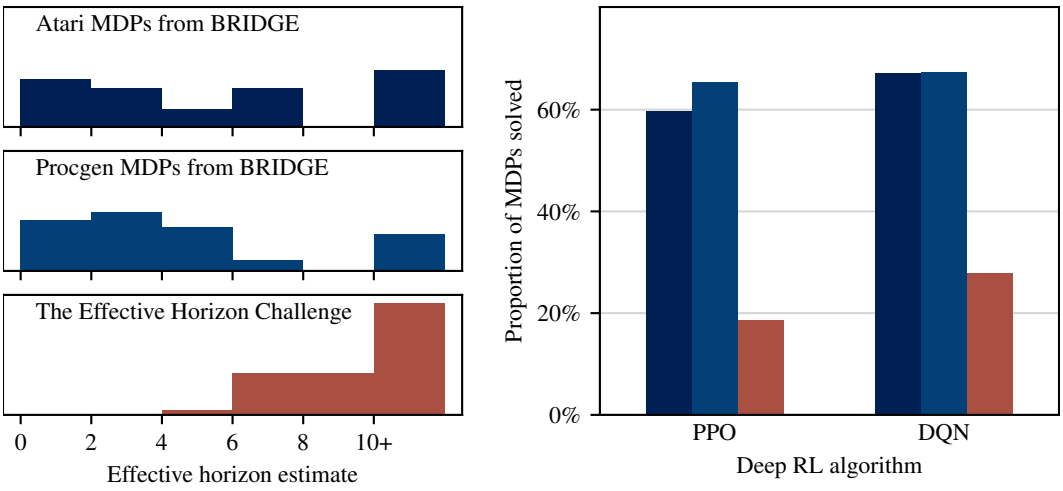

Figure 1: We introduce the Effective Horizon Challenge (EHC), a new deep RL benchmark that consists of MDPs with longer *effective horizons* than those in past work. The effective horizon measures how easy it is to solve an MDP with Monte Carlo lookahead search, and previous benchmarks like the Atari games in the ALE have low effective horizon (top left corner). In contrast, the EHC environments have much longer effective horizons (lower left). We find that the deep RL algorithms PPO and DQN struggle in the EHC, solving only around 20% of the environments. This shows that RL struggles in settings with long effective horizons and suggests that future work on deep RL should focus on solving these harder environments.

Our findings suggest that while deep RL has produced impressive results, its success may still be limited to environments with short effective horizons. In other words, PPO and DQN seem to struggle to explore in cases where random rollouts cannot be used to select optimal actions. The MDPs in the EHC are simple in many ways—they are deterministic, they all have fewer than four million states (and most less than one million), they do not have extremely sparse rewards—and yet popular algorithms still struggle to solve them. We hope that the EHC serves as a catalyst to develop improved RL algorithms that can leverage more advanced exploration strategies to succeed beyond the short effective horizon regime.

2 Background

Before introducing the EHC, we survey related work and present background information on MDPs, RL, and the effective horizon.

2.1 Related work

Deep RL benchmarks Benchmarking has played a central role in the progress of deep RL. In discrete-action RL, the Arcade Learning Environment (ALE) [1] is a widely used testbed that emphasizes performance across diverse games. Procgen [9] is a newer benchmark which has been used to test generalization in RL. However, recent work by Laidlaw et al. [7, 8] shows that many environments in these benchmarks can be solved by acting greedily with respect to the Q-function of the random policy. This raises the concern that improvements in sample efficiency in these benchmarks may not generalize to deep RL solving fundamentally harder decision-making problems.

Some benchmarks such as bsuite [11], MDP Playground [12], SEGAR [13], MiniGrid [**?**], and NetHack [14] are designed to empirically evaluate deep RL algorithms across various axes of environment difficulty. **bsuite** [11] decomposes RL performance into interpretable components such as exploration, reward sparsity, and memory, but its tasks are typically low-dimensional and analytically simple. **SEGAR** [13] provides a flexible 3D physics sandbox to test inductive biases and reasoning, though its continuous control setting and partial observability make it difficult to isolate specific planning challenges. **MiniGrid** [**?**] is a gridworld-based benchmark that emphasizes exploration and memory, but the environments are small, not tabularized, and lack formal complexity metrics, limiting theoretical analysis. **MDP Playground** [12] offers configurable MDPs with explicit control over factors like stochasticity and reward delays, but does not focus on long-horizon planning

or exploration. **NetHack** [14] is a more semantically rich and procedurally complex benchmark, but it is hard to analyze theoretically due to its massive action space, textual interface, and lack of known optimal policies.

These platforms allow testing across a wide range of environment properties, but often lack theoretical grounding for why some tasks are harder than others. BRIDGE [7] partially fills this gap by introducing tabular versions of existing environments, enabling exact Q-function computation and introducing the effective horizon as a complexity metric; however, most environments in BRIDGE have short effective horizons, so it cannot be easily used to benchmark deep RL performance in MDPs with long effective horizons. Furthermore, the MDPs in bsuite, SEGAR, and MDP Playground are quite artificial; they are specifically constructed to have a certain structure. Thus, it is unclear if they represent difficult sequential decision-making problems more broadly. Of course many other RL benchmarks are also artificial or "toy" in some sense—they are often literally games—but at least they are based on some previously existing interesting decision problems inspired by real-world tasks like navigation, balancing competing objectives, and hand-eye coordination.

In this work, we introduce a diverse dataset of MDPs, the Effective Horizon Challenge, which both have long effective horizons (unlike Atari games), are not too artificial (unlike bsuite, SEGAR, and MDP Playground), and include tabular representations to enable theoretical analysis. Unlike prior benchmarks focused on improving sample efficiency within already-solvable environments (e.g., the Atari 100K challenge [15]), our benchmark explicitly tests whether algorithms can solve previously *unsolvable* tasks using principled exploration. The EHC shares a key strength with ALE and Procgen: a diversity of environments that prevents overfitting to a single task. However, it moves beyond them by focusing on environments that demand deep planning, making it a more challenging and informative benchmark for the next generation of RL algorithms.

2.2 Setting and notation

We consider the standard reinforcement learning (RL) setting in a tabular, deterministic, episodic Markov decision process (MDP). Our benchmark focuses on discrete-action environments because the effective horizon is only defined for such MDPs; furthermore, discrete action MDPs support the widest range of deep RL algorithms, unlike continuous control environments which require more specialized algorithms. We consider deterministic environments to further simplify the benchmark, but the MDPs in the EHC could be modified in the future by adding, for example, sticky actions [16] to make them stochastic; Laidlaw et al. [8] define an equivalent notion of effective horizon for stochastic environments.

An MDP consists of a finite set of states $\mathcal{S}$, a finite action space $\mathcal{A}$, a horizon $T \in \mathbb{N}$, a start state s_1, a deterministic transition function $f : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$, a reward function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and a discount factor $\gamma \in [0, 1]$.

An RL agent interacts with an MDP over multiple episodes, each beginning from a fixed initial state s_1. During each episode, at every timestep $t \in [T]$ (where $[n] = \{1, \dots, n\}$), the agent observes the current state s_t, selects an action a_t, receives a reward $R(s_t, a_t)$, and transitions to the next state $s_{t+1} = f(s_t, a_t)$. A policy π is defined as a set of functions $\pi_1, \dots, \pi_t : \mathcal{S} \to \Delta(\mathcal{A})$, specifying a probability distribution $\pi_t(a \mid s)$ over actions for each state and timestep. When the policy is deterministic at a given state, we slightly abuse notation and write $a = \pi_t(s)$ to denote the action selected by π_t in state s.

For each timestep $t \in [T]$, the Q-function and value function of a policy are defined by

$$Q_t^\pi(s, a) = \mathbb{E}_\pi \left[\sum_{t'=t}^{T} \gamma^{t'-t} \, R(s_{t'}, a_{t'}) \mid s_t = s, a_t = a \right]$$
$$V_t^\pi(s) = \mathbb{E}_\pi \left[\sum_{t'=t}^{T} \gamma^{t'-t} \, R(s_{t'}, a_{t'}) \mid s_t = s \right].$$

The objective of an RL algorithm is to find an optimal policy π^*, which maximizes $J(\pi) = V_1^\pi(s_1)$, the expected discounted sum of rewards over an episode, also known as the return of the policy π.

Generally, an RL algorithm can be run for any number of timesteps n (i.e., counting one episode as T timesteps), returning a policy π^n. We define the *sample complexity* N of an RL algorithm as the minimum number of timesteps needed such that the algorithm has at least a 50-50 chance of returning

an optimal policy:

$$N = \min \{n \in \mathbb{N} \mid \mathbb{P}(J(\pi^n) = J^*) \geq 1/2\}.$$

Here, the probability is with respect to any randomness in the algorithm itself. One can estimate the sample complexity N empirically by running an algorithm several times, calculating the number of samples n needed to reach the optimal policy during each run, and then taking the median.

The effective horizon and GORP A key concept introduced by Laidlaw et al. [7] is the *effective horizon*, a complexity measure that helps explain when deep RL with random exploration succeeds or fails. This is motivated by the surprising empirical finding that in many benchmark environments, the optimal policy can be recovered by acting greedily with respect to the Q-function of the *random* policy (i.e., $\pi_t^{\text{rand}}(a \mid s) = 1/|\mathcal{A}|$). In such environments, a single step of policy iteration—starting from the random policy—can suffice to find the optimal policy.

Laidlaw et al. [7] extend this property to define an MDP as *k-QVI-solvable* if applying $k-1$ steps of Q-value iteration starting from the random policy yields a Q-function Q^k such that any greedy policy w.r.t. Q^k is optimal, where the set of greedy policies $\Pi(Q)$ for a Q-function Q is defined as

$$\Pi(Q) = \left\{\pi \mid \forall s, t \quad \pi_t(s) \in \arg\max_{a \in \mathcal{A}} Q_t(s, a)\right\}.$$

Definition 2.1 (k-QVI-solvable)**.** *Let $Q^1 = Q^{\pi^{rand}}$ and Q^{i+1} be the result of applying one step of Q-value iteration to Q^i for $i = 1, \ldots, T-1$, i.e.,*

$$Q_t^{i+1}(s, a) = R_t(s, a) + \arg\max_{a' \in \mathcal{A}} Q_{t+1}^i(f(s, a), a'). \tag{1}$$

We say an MDP is k-QVI-solvable *for some $k \in [T]$ if every policy in $\Pi(Q^k)$ is optimal.*

In other words, a small number of Q-value iteration steps on the Q-function of a random (or fixed) policy may suffice to uncover the optimal policy.

To efficiently solve such environments, Laidlaw et al. [7] propose the *Greedy Over Random Policy* (GORP) algorithm. GORP simulates k steps of lookahead search and estimates $Q^{\pi^{\text{rand}}}$ using Monte Carlo rollouts. Crucially, GORP separates exploration (through random rollouts) from learning, making it both practically effective and theoretically analyzable. The resulting complexity measure, the *effective horizon H*, combines the number of QVI steps k with the number of rollouts m required to accurately estimate Q-values at the leaf nodes:

Definition 2.2 (Effective horizon)**.** *Given $k \in [T]$, let $H_k = k + \log_A m_k$, where m_k is the minimum value of m needed for GORP with parameter k to return the optimal policy with probability at least $1/2$, or ∞ if no value of m suffices. The* effective horizon *is $H = \min_k H_k$.*

The sample complexity of GORP is then bounded by $O(T^2 A^H)$. H is called the effective horizon because GORP's sample complexity is exponential only in the H, while a randomly-exploring algorithm's worst-case sample complexity is exponential in the full horizon T.

[7] introduce a dataset called BRIDGE of MDPs based on the ALE and Procgen, and find that many of these MDPs have short effective horizons. In particular, about two thirds of the MDPs in BRIDGE are 1-QVI-solvable, meaning they can be solved using simply by acting greedily with respect to $Q^{\pi^{\text{rand}}}$. This suggests that these existing benchmarks cannot be used to test whether deep RL succeeds beyond the short effective horizon regime. We aim to fill this gap by introducing the EHC.

A note on discount factors One gap between the effective horizon definition and typical RL practice is the use of discount factors γ. Most RL papers *train* with a discount factor $\gamma < 1$, which is often treated as a hyperparameter to tune, but then *evaluate* with no discounting, i.e. $\gamma = 1$. For the EHC we follow the typical practice of using $\gamma = 0.99$ during training but $\gamma = 1$ during evaluation. This makes our analysis of the effective horizon slightly different than Laidlaw et al. [7]; we define the effective horizon based on the GORP parameters m and k that allow GORP using $\gamma = 0.99$ to find an optimal policy in the undiscounted MDP. We furthermore confirm in all environments that the optimal policy for $\gamma = 0.99$ is also optimal for $\gamma = 1$.

3 The Effective Horizon Challenge

The Effective Horizon Challenge consists of 43 MDPs designed to have long effective horizons, allowing us to test whether deep RL can succeed in such environments. The MDPs are based on a number of classical planning environments and two-player games and then filtered to ensure that they are of sufficient difficulty to constitute a challenge for deep RL algorithms.

Dataset desiderata We aimed to make the EHC satisfy a few desiderata. First, we didn't want the MDPs in the EHC to be too artificial. Thus, instead of constructing MDPs from scratch like in other benchmarks [11, 13, 12], we draw from existing sequential decision making problems in the literature.

Second, we wanted the MDPs in the EHC to have long effective horizons, but not to be very difficult along other measures of difficulty. For example, Osband et al. [11] identify aspects of RL difficulty including memory, noise, generalization, and so on. We thus designed the MDPs in the EHC to have small state spaces (up to 4 million states), short horizons (up to 100 timesteps), no stochasticity, and non-sparse reward functions. By focusing on making our MDPs only hard in terms of effective horizon, we can isolate how well deep RL can solve long effective horizon environments without other factors confounding the analysis.

Finally, we wanted the EHC to enable theoretical as well as empirical analysis, allowing researchers to explore assumptions that might hold in realistic environments that could enable efficient RL. Thus, similarly to Laidlaw et al. [7], we construct full tabular representations of each MDP in the EHC. This has additional benefits beyond theoretical analysis: it makes the environments extremely fast to run and easy to implement in any programming language, as the environment step can be reduced to a few lookups in the tabular transition and reward matrices.

Sources of MDPs To create the EHC, we constructed MDPs based on four classical planning environments (3×3 slide tile puzzles, towers of Hanoi, the $2 \times 2 \times 3$ "tower" Rubik's cube, and Sokoban levels) as well as four two-player games (4×4 Go, miniature chess played on 4×3 and 4×4 boards, checkers/draughts on 5×5 and 6×4 boards, and Connect Four played on 6×5 and 7×4 boards). See Figure 2 for illustrations of these MDPs.

The classical planning environments are based on planning problems described in Planning Domain Definition Language (PDDL) [17], which encodes goals as a set of logical predicates which must all be satisfied. For example, the goal in Sokoban environments consists of predicates indicating each box is on a target square, and the goal in Rubik's cube consists of predicates indicating the colors are in the correct locations. Our reward functions for these environments are based on the number of logical predicates in the goal satisfied before and after an action is taken. In particular, letting the potential $\Phi(s)$ represent the number of goal predicates satisfied in state s, the reward function is $R(s, a) = \Phi(s') - \Phi(s)$, where s' is the state reached from taking action a in state s. This ensures that the maximum reward is achieved by completing the entire goal, but also gives reward for completing parts of the goal. For each planning environment, we constructed several MDPs by varying the start state (for Rubik's cube and slide tile), problem size (for Hanoi), or layout (for Sokoban).

For the games, we first constructed a full tabular representation of the two-player game. The reward function for each game assigns 1 for a win, -1 for a loss, and 0 for a draw. We also added shaped rewards for some of the checkers, chess, and Go MDPs. These shaped rewards are based on a potential function for each game that measures how much the current player is winning or losing: the Tromp-Taylor score for Go and the difference in piece values for chess and checkers (for checkers, we assign a king double the value of a regular piece). To use these games as single-player environments for benchmarking purposes, we then fixed deterministic opponent policies. The opponent policies are constructed by sampling an action at each state from a Boltzmann distribution based on either the optimal or noisy-optimal Q-value for the opponent at that state, assuming optimal or noisy-optimal play by the RL agent. We sampled opponent policies at multiple temperatures with and without reward shaping. We filtered out cases where the opponent policies were so strong that they made it impossible for the first player to win.

Filtering We expected that many of our MDPs would have longer effective horizons due to the difficult strategies required for these planning problems and games. However, we suspected that some

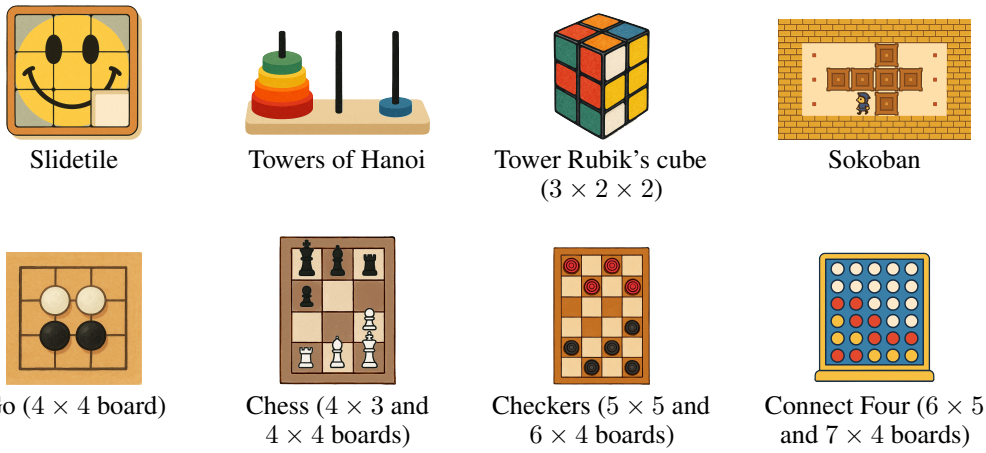

Figure 2: The 43 MDPs in the Effective Horizon Challenge include classical planning problems and two-player games against a fixed opponent. We calculate the full tabular representation of each MDP, necessitating smaller-scale versions of many of the games. See Section 3 for more details.

might still be too easy to solve. Thus, we removed MDPs that could be solved either via GORP or random guessing in fewer than ten million timesteps, since this is how many samples we used for deep RL algorithms (see Section 4). To calculate whether GORP could solve each MDP, we ran it empirically, exhaustively searching over GORP's first parameter k and then performing binary search on its second parameter m to see if it could achieve the optimal reward; if any combination of parameters could solve an MDP in fewer than ten million timesteps we excluded that MDP.

To determine whether random guessing could solve each MDP, we used the tabular representation to calculate the probability that a random sequence of actions from the initial state would achieve optimal reward; denote this as p_{opt}. Then, it is simple to show that an RL algorithm which samples $\log(2)/p_{\text{opt}}$ episodes of random actions has at least a $1/2$ chance of seeing one action sequence with the optimal reward, which it can return as an optimal policy (since the MDP is deterministic, a closed-loop policy is not needed). Thus, we calculated $T \log(2)/p_{\text{opt}}$ as an upper bound on the sample complexity of this random guessing algorithm for each MDP; we removed those for which the bound was below ten million.

After removing MDPs which could be solved by either GORP or random guessing in fewer than ten million timesteps, we were left with 43 MDPs: 28 based on games and 15 based on classical planning problems.

Analyzing the effective horizon To make sure our dataset does have longer effective horizons than past deep RL benchmarks, we measure the effective horizons of our MDPs and other popular environments. Since it is intractable to exactly compute the effective horizon of full Atari games or other deep RL benchmarks, we instead use the horizon-limited tabular versions of Atari and Procgen [9] environments in the BRIDGE dataset [7]. We approximate the effective horizon for each MDP by tuning GORP over many possible values of its parameters k and m. Then, we use the formula for the effective horizon $H = k + \log_A m$ from Laidlaw et al. [7], where k and m are selected to make H as small as possible while still allowing GORP to find an optimal policy.

The effective horizons of MDPs in the EHC versus past benchmarks are shown in Figure 1. We find that the effective horizon is less than 8 in 72% and 82% of the Atari and Procgen MDPs from BRIDGE, respectively. In contrast, only 23% of MDPs in the Effective Horizon Challenge have an effective horizon less than 8. This analysis confirms that the EHC is a useful benchmark for measuring deep RL performance in environments with long effective horizons.

4 Experiments

To measure whether popular deep RL algorithms can solve the environments in the Effective Horizon Challenge, we run PPO and DQN in each MDP. We use the Stable Baselines3 [18] implementations

Benchmark	MDPs solved PPO	DQN
Atari MDPs in BRIDGE	60%	67%
Procgen MDPs in BRIDGE	65%	67%
The Effective Horizon Challenge (ours)	19%	28%

Table 1: While deep RL algorithms solve most Atari and Procgen MDPs, they struggle in our dataset of MDPs with longer effective horizons. The Effective Horizon Challenge consists of 43 MDPs, while BRIDGE [7] includes 67 Atari MDPs and 55 Procgen MDPs. See Section 4 for experiment details.

of both algorithms and run them for ten million timesteps across three random seeds and 2-3 hyperparameter settings. Every 1,000 timesteps during training, we measure the reward achieved by a deterministic version of the current policy (i.e., always taking the action with the highest Q-value or action probability). If any hyperparameter setting enables the algorithm to reach the optimal policy during training, we consider the MDP solved by that algorithm. For PPO, we tune the number of steps sampled in each environment within the set $\{128, 1280, 12800\}$. For DQN, we tune the epsilon-greedy schedule to decay from $\epsilon = 1$ to $\epsilon = 0.01$ either over the first 10% of training or over the entire course of training. Besides these, we use the default Atari hyperparameters for each algorithm.

The results of our experiments are shown in Figure 1 and Table 1, where we also run each deep RL algorithm on the Atari and Procgen MDPs from the BRIDGE dataset. We find that both PPO and DQN struggle to solve the long-effective-horizon MDPs in the EHC: DQN solves 28%, while PPO solves only 19%. In contrast, the algorithms solve 60-70% of the Atari and Procgen MDPs in BRIDGE. This suggests that the algorithms do struggle to solve environments with long effective horizons.

Figure 3 in Appendix A shows the learning curves for each algorithm in each MDP in the EHC. In virtually all MDPs, PPO and DQN are able to increase reward at the beginning of training, but they often appear to then get caught in local maxima of reward, where the reward plateaus and fails to keep increasing. This could suggest that they are behaving similarly to GORP by myopically optimizing with a few steps of lookahead, rather than successfully exploring the environment to find a globally optimal policy.

5 Conclusion and future work

We have introduced a novel benchmark, the Effective Horizon Challenge, for reinforcement learning focused on environments with a long *effective horizon*, a measure of MDP complexity and exploration difficulty introduced by Laidlaw et al. [7]. The EHC fills a gap in the existing landscape of RL benchmarks, whose environments generally have short effective horizons, making exploration trivial. Although the MDPs in the EHC are deterministic and have small state spaces, we find RL fails to find an optimal policy in them.

Our results suggest that hill-climbing on current benchmarks like the ALE can lead to improved efficiency in short effective horizon environments, but may not enable RL to solve fundamentally harder MDPs like those in the EHC. Improving performance in the EHC may require new exploration techniques or fundamental insights into how to structure RL to efficiently learn in environments that cannot be solved in a few steps of lookahead on the random policy's Q-function.

While our results of evaluating deep RL algorithms in the EHC are striking, they are limited to older RL algorithms and a limited amount of environment interaction. In the future, we plan to address both of these limitations by testing more RL algorithms and by using more environment samples. Regardless, it is still striking that two of the most-used deep RL algorithms fail to solve environments with relatively short episodes and small state spaces even after ten million timesteps. We hope our benchmark kick-starts a push towards more effective exploration in deep RL that can efficiently solve long effective-horizon decision problems.

Acknowledgments and Disclosure of Funding

This work was supported by a grant from Open Philanthropy to the Center for Human-Compatible Artificial Intelligence at UC Berkeley and a grant from the National Science Foundation (NSF) Human-Centered Computing (HCC) to Professor Anca Dragan (award number 2310757). Cassidy Laidlaw is supported by an Open Philanthropy AI Fellowship.

References

[1] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The Arcade Learning Environment: An Evaluation Platform for General Agents. *Journal of Artificial Intelligence Research*, 47: 253–279, June 2013. ISSN 1076-9757. doi: 10.1613/jair.3912. URL https://www.jair.org/index.php/jair/article/view/10819.

[2] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015. ISSN 1476-4687. doi: 10.1038/nature14236. URL https://www.nature.com/articles/nature14236. Bandiera_abtest: a Cg_type: Nature Research Journals Number: 7540 Primary_atype: Research Publisher: Nature Publishing Group Subject_term: Computer science Subject_term_id: computer-science.

[3] Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining Improvements in Deep Reinforcement Learning, October 2017. URL http://arxiv.org/abs/1710.02298. arXiv:1710.02298 [cs].

[4] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering Atari, Go, Chess and Shogi by Planning with a Learned Model. *Nature*, 588(7839):604–609, December 2020. ISSN 0028-0836, 1476-4687. doi: 10.1038/s41586-020-03051-4. URL http://arxiv.org/abs/1911.08265. arXiv:1911.08265 [cs, stat].

[5] Weirui Ye, Shaohuai Liu, Thanard Kurutach, Pieter Abbeel, and Yang Gao. Mastering Atari Games with Limited Data. *arXiv:2111.00210 [cs]*, October 2021. URL http://arxiv.org/abs/2111.00210. arXiv: 2111.00210.

[6] Max Schwarzer, Johan Samir Obando-Ceron, Aaron C. Courville, Marc G. Bellemare, Rishabh Agarwal, and Pablo Samuel Castro. Bigger, Better, Faster: Human-level Atari with human-level efficiency. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 30365–30380. PMLR, 2023. URL https://proceedings.mlr.press/v202/schwarzer23a.html.

[7] Cassidy Laidlaw, Stuart Russell, and Anca Dragan. Bridging RL Theory and Practice with the Effective Horizon. In *Advances in Neural Information Processing Systems*, 2023. URL http://arxiv.org/abs/2304.09853. arXiv:2304.09853 [cs, stat].

[8] Cassidy Laidlaw, Banghua Zhu, Stuart Russell, and Anca Dragan. The Effective Horizon Explains Deep RL Performance in Stochastic Environments. arXiv, April 2024. doi: 10.48550/arXiv.2312.08369. URL http://arxiv.org/abs/2312.08369. arXiv:2312.08369 [stat].

[9] Karl Cobbe, Christopher Hesse, Jacob Hilton, and John Schulman. Leveraging Procedural Generation to Benchmark Reinforcement Learning, July 2020. URL http://arxiv.org/abs/1912.01588. arXiv:1912.01588 [cs, stat].

[10] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms, August 2017. URL http://arxiv.org/abs/1707.06347. arXiv: 1707.06347.

[11] Ian Osband, Yotam Doron, Matteo Hessel, John Aslanides, Eren Sezener, Andre Saraiva, Katrina McKinney, Tor Lattimore, Csaba Szepesvári, Satinder Singh, Benjamin Van Roy, Richard S. Sutton, David Silver, and Hado van Hasselt. Behaviour Suite for Reinforcement Learning. In *8th International Conference on Learning Representations*. OpenReview.net, 2020. URL https://openreview.net/forum?id=rygf-kSYwH.

[12] Raghu Rajan, Jessica Lizeth Borja Diaz, Suresh Guttikonda, Fabio Ferreira, André Biedenkapp, Jan Ole Von Hartz, and Frank Hutter. MDP Playground: An Analysis and Debug Testbed for Reinforcement Learning. *Journal of Artificial Intelligence Research*, 77:821–890, July 2023. ISSN 1076-9757. doi: 10.1613/jair.1.14314. URL https://www.jair.org/index.php/jair/article/view/14314. Publisher: AI Access Foundation.

[13] R. Devon Hjelm, Bogdan Mazoure, Florian Golemo, Felipe Frujeri, Mihai Jalobeanu, and Andrey Kolobov. The Sandbox Environment for Generalizable Agent Research (SEGAR), March 2022. URL http://arxiv.org/abs/2203.10351. arXiv:2203.10351 [cs].

[14] Heinrich Küttler, Nantas Nardelli, Alexander H. Miller, Roberta Raileanu, Marco Selvatici, Edward Grefenstette, and Tim Rocktäschel. The NetHack Learning Environment, December 2020. URL http://arxiv.org/abs/2006.13760. arXiv:2006.13760 [cs].

[15] Lukasz Kaiser, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski, Roy H. Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, Afroz Mohiuddin, Ryan Sepassi, George Tucker, and Henryk Michalewski. Model-Based Reinforcement Learning for Atari, April 2024. URL http://arxiv.org/abs/1903.00374. arXiv:1903.00374 [cs].

[16] Marlos C. Machado, Marc G. Bellemare, Erik Talvitie, Joel Veness, Matthew Hausknecht, and Michael Bowling. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *Journal of Artificial Intelligence Research*, 61:523–562, 2018. URL https://www.jair.org/index.php/jair/article/view/11182.

[17] D. McDermott, M. Ghallab, A. Howe, Craig A. Knoblock, A. Ram, M. Veloso, Daniel S. Weld, and D. Wilkins. PDDL-the planning domain definition language. 1998. URL https://www.semanticscholar.org/paper/PDDL-the-planning-domain-definition-language-McDermott-Ghallab/d82c6b8081343b2eae63d45feefe630233ad60e1.

[18] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *The Journal of Machine Learning Research*, 22(1):12348–12355, 2021. URL https://dl.acm.org/doi/abs/10.5555/3546258.3546526. Publisher: JMLRORG.

Appendix

A Additional results

See the following page for the learning curves of PPO and DQN in all MDPs in the EHC.

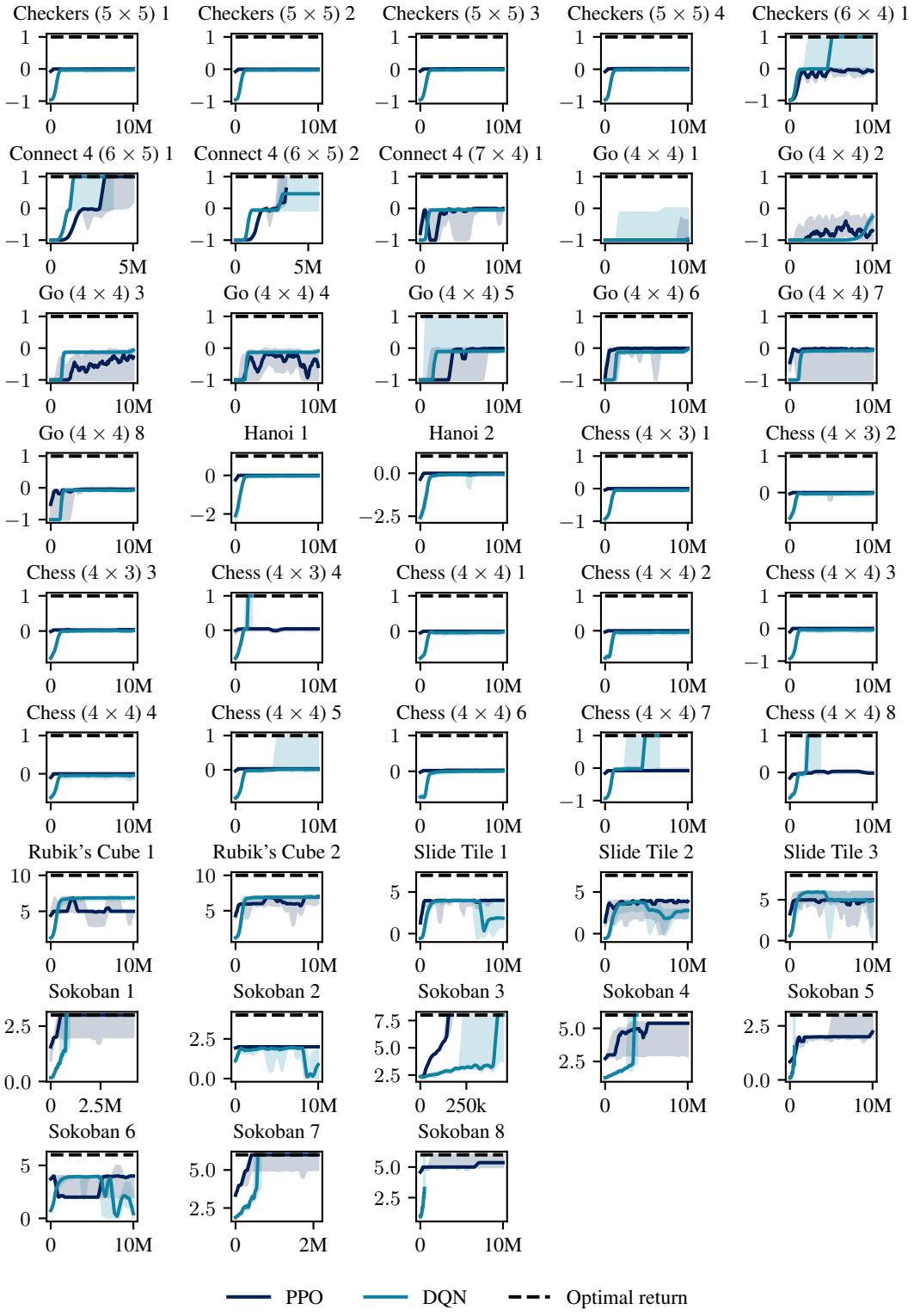

Figure 3: Learning curves for PPO and DQN across MDPs in the EHC. The shaded region shows the range across random seeds.