Efficient Exploration of Zero-Sum Stochastic Games

Carlos Martin,[1] **Tuomas Sandholm**[1,2,3,4]
[1]Carnegie Mellon University
[2]Strategy Robot, Inc.
[3]Optimized Markets, Inc.
[4]Strategic Machine, Inc.
cgmartin@cs.cmu.edu, sandholm@cs.cmu.edu

Abstract

We investigate the increasingly important and common game-solving setting where we do not have an explicit game description but only oracle access through playing, such as in financial or military simulations and computer games. During a limited-duration learning phase, the algorithm can control the actions of both players to try to learn the game and how to play it well. After that, the algorithm has to produce a policy that plays well against *any* opponent. For the stochastic game setting (and normal-form games as a special case), we propose using the distribution of state-action value functions induced by a belief distribution over possible environments in various ways. We compare the performance of many exploration policies for this task, including generalizations of exploration policies from the single-agent setting to games, as well as new ones. The multiagent setting raises new challenges and qualitative changes in behavior. Experiments across different environments show orders of magnitude sample-efficiency improvements over a well-known model-free approach. Modified versions of Thompson sampling and UCB are consistently among the most sample efficient. The former is also computationally efficient.

Introduction

We study how to efficiently explore zero-sum games whose payoffs and dynamics are initially unknown. The agent is given a certain number of episodes to learn as much useful information about the game as possible. During this learning, the agent can control the play of both players in the game and the rewards obtained are fictional and thus do not count toward the evaluation of the final strategy. After this exploration phase, the agent must recommend a strategy for playing the game that should be minimally exploitable by an adversary (who has complete knowledge of the environment and can thus play optimally against it). This setup is called *pure exploration* in single-agent *reinforcement learning (RL)*. This setup is important for simulation-based games in which a black-box simulator is queried with strategies to obtain samples of the players' resulting utilities (Vorobeychik and Wellman 2009), as opposed to the rules of the game being explicitly given. For example, in many military settings, war game simulators are used to generate strategies, and then the strategies need to be ready to deploy in case of actual war (Marchesi, Trovò, and Gatti 2019). Another example is finance, where trading strategies are generated in simulation, and then they need to be ready for live trading. A third example is video games such as Dota 2 (Berner et al. 2019) and Starcraft II (Vinyals et al. 2019), where AIs can be trained largely through self play, but most prior techniques have not focused on exploration approaches that yield strategies that have low exploitability against *every possible* adversary strategy.

This raises the challenge not only of learning approximate equilibria with noisy observations, but also of learning in as few queries as possible, since running the simulator is usually expensive. Prior work on games such as Dota 2 and Starcraft II has not evaluated the exploitability of the learned strategies. In this paper, we experiment on games that are small enough that game-theoretic exploitability of the recommended strategy can be evaluated. We study exploration in two-player zero-sum stochastic games, which subsume two-player zero-sum extensive-form games that are otherwise of perfect-information but have simultaneous moves. Our approach is *model driven*: our algorithms incrementally build a model of the game and uses it to guide exploration.

Related research

In single-agent RL, Q-learning (Watkins and Dayan 1992) learns state-action values directly in a model-free way, that is, without learning the structure of the environment. Dearden, Friedman, and Russell (1998) extended it to incorporate uncertainty by propagating probability distributions over the Q values in order to compute a myopic approximation of the value of information, which measures the expected improvement in future decision quality that can be gained from exploration. Bellemare, Dabney, and Munos (2017) and O'Donoghue et al. (2018) argue for the importance of the Q value distribution and propose a new version of Bellman updating that incorporates uncertainty.

A similar problem has been studied in single-agent model-based RL. *Posterior sampling RL* (Osband and Roy 2017; Agrawal and Jia 2017) samples an environment from the agent's belief distribution, follows a policy that is optimal with respect to it, and then updates the agent's beliefs about the environment with the resulting observations. Zhou, Li, and Zhu (2020) extended it to zero-sum imperfect-

information extensive-form games and presented an algorithm that converges to a Nash equilibrium at a bounded rate.

Even in single-agent settings, sampling a new policy on every step within an episode is inefficient because it does not do *deep exploration*, which accounts not only for information gained by taking an action but also for how the action may position the agent to more effectively acquire information later (Russo et al. 2018). In deep exploration, a single policy is chosen at the beginning of each episode and followed for its duration. One approach to deep exploration chooses actions that are optimal with respect to a value function that is sampled from an ensemble (Osband et al. 2016, 2019). Each element of the ensemble is a deep neural network trained with deep Q-learning (Mnih et al. 2015), and the ensemble constitutes a belief distribution over possible value functions of the environment. It incentivizes experimentation with actions of uncertain value because uncertainty induces variance in the sampled value estimate. Chen et al. (2017) also use an ensemble of Q functions but, instead of sampling from them, use the resulting upper confidence bounds. Mavrin et al. (2019) combine a decaying schedule with exploration bonuses computed from upper quantiles of the learned distribution. Littman (1994) describes a Q-learning-like algorithm for finding optimal policies for one player in stochastic games (Shapley 1953) when playing against an opponent that the algorithm does not control.

Sandholm and Crites (1996) study RL in a repeated game. They study the role of other agents making the setting stochastic for a learner, the role of exploration, and convergence to cycles of different lengths, and how recurrent neural networks can, in principle, help with those issues. Claus and Boutilier (1998) study RL in cooperative settings, showing that several optimistic exploration strategies increase the likelihood of reaching an optimal equilibrium. Wang and Sandholm (2002) describe an algorithm that converges almost surely to an optimal equilibrium in any team stochastic game. Conitzer and Sandholm (2003) present BL-WoLF, a framework for learnability in repeated zero-sum games where the cost of learning is measured by the losses the learning agent accrues, and guaranteed learnability results for families of games. Hu and Wellman (2003) present Nash Q-learning for general-sum stochastic games, which is guaranteed to converge to an equilibrium if all agents follow the algorithm and the stage games satisfy certain highly restrictive conditions. Ganzfried and Sandholm (2009) design algorithms for computing equilibria in special classes of stochastic games of imperfect information. Heinrich and Silver (2016) introduced an approach to learning approximate Nash equilibria without prior domain knowledge by combining fictitious self-play with deep RL. Lanctot et al. (2017) introduce a meta-algorithm for *multiagent RL (MARL)* based on approximate best responses to mixtures of policies generated using deep RL. Srinivasan et al. (2018) apply candidate policy update rules to model-free MARL in adversarial sequential decision problems, showing empirical convergence to approximate Nash equilibria in self play. Casgrain, Ning, and Jaimungal (2019) study Nash Q-learning but they use a neural network to model the Q function, decomposing it into a sum of the state value function and a specific form of action advantage function. Sokota, Ho, and Wiedenbeck (2019) use neural networks to learn a mapping from mixed-strategy profiles to deviation payoffs in order to approximate role-symmetric equilibria in large simulation-based games. Lockhart et al. (2019) present exploitability descent, an algorithm to compute approximate equilibria in extensive-form games by direct policy optimization against worst-case opponents.

Our work also relates to the multi-armed bandit problem. In the standard version, the agent must—over a given period of time—acquire information about the mean payoff of each action while simultaneously trying to maximize the cumulative payoff. Because of this tradeoff, multi-armed bandit problems exemplify the exploration-exploitation dilemma. In the *pure exploration* version, there is a learning phase first, during which the rewards obtained are fictional and do not count toward the evaluation (Bubeck, Munos, and Stoltz 2009). Then the agent recommends an arm (that is, action) to play going forward. The agent's performance is measured purely by the effectiveness of her recommended action. This performance measure is called *simple regret*, in contrast to the cumulative regret of the standard problem where rewards throughout the process count. Our paper focuses on the multiagent generalization of pure exploration.

Garivier, Kaufmann, and Koolen (2016) study the problem of pure exploration in the context of a *sequential*-move game with the aim of identifying an ε-maxmin action with probability at least $1 - \delta$. Marchesi, Trovò, and Gatti (2019) tackle the problem of learning equilibria in simulation-based games of infinite strategy spaces with high confidence using as few simulator queries as possible. They propose an algorithm for the fixed-confidence setting (guaranteeing the desired confidence level while minimizing the number of queries) and one for the fixed-budget setting (maximizing the confidence without exceeding the given maximum number of queries).

Zero-sum stochastic games

$\triangle\mathcal{X} \subseteq \mathcal{X} \to \mathbb{R}$ denotes the set of all probability distributions on a set $\mathcal{X}$. $\mathcal{N}$ denotes a set of players. $i : \mathcal{N}$ denotes a player. $-i$ denotes the remaining player(s). $\mathcal{S}$ denotes a set of states. $s : \mathcal{S}$ denotes a state. $\mathcal{A}_i(s)$ denotes i's set of actions in s. $a_i : \mathcal{A}_i(s)$ denotes i's action in s. $a : \prod_i \mathcal{A}_i(s)$ denotes an action profile in s. $\sigma_i : \triangle\mathcal{A}_i(s)$ denotes i's strategy in s. $\sigma : \prod_i \triangle\mathcal{A}_i(s)$ denotes a strategy profile in s. $\pi_i : \prod_s \triangle\mathcal{A}_i(s)$ denotes i's policy. $\pi : \prod_i \prod_s \triangle\mathcal{A}_i(s)$ denotes a policy profile. $r_i : \prod_s (\prod_i \mathcal{A}_i(s) \to \mathbb{R})$ denotes i's reward function. $\delta : \prod_s (\prod_i \mathcal{A}_i(s) \to \triangle\mathcal{S})$ denotes a transition function. $\gamma : [0, 1]$ denotes a discount factor. $s_{\text{init}} : \mathcal{S}$ denotes the initial state.

A stochastic game is a tuple $(\mathcal{N}, \mathcal{S}, \mathcal{A}, r, \delta, \gamma, s_{\text{init}})$. We consider *finite-horizon* games, so $\gamma = 1$, the transition graph induced by δ is acyclic, and terminal states have $\mathcal{A}_i(s) = \varnothing$. The game proceeds as a sequence of *stage games* in which all players choose actions simultaneously, each player receives a reward, and the game transitions to a new state. This process repeats until a terminal state is reached. We consider two-player zero-sum games, so $\mathcal{N} = \{1, 2\}$ and $\sum_i r_i = 0$.

This model is quite general and captures many types of

games. If there is a single nonterminal state, we have a matrix game (Sandholm and Crites 1996). If $|\mathcal{A}_{-i}(s)| = 1$ for every $s : \mathcal{S}$, we have a Markov decision process (MDP) for i. If both of the above conditions hold, we have a multi-armed bandit. If the state transition graph induced by δ is a tree, we have a perfect-information extensive-form game.

Policy recommendation under uncertainty

Let $u_i(g, \pi)$ denote the expected return to player i of policy profile π under game g. Suppose we face the task of recommending a policy for player i, under the assumption that our opponent will play optimally against it. To maximize our expected utility, we should recommend

$$\operatorname*{argmax}_{\pi_i} \min_{\pi_{-i}} u_i(g, \pi) \tag{1}$$

We define the *regret* incurred by $\hat{\pi}_i$ under g as

$$\max_{\pi_i} \min_{\pi_{-i}} u(g, \pi) - \min_{\pi_{-i}} u(g, \hat{\pi}_i, \pi_{-i}) \tag{2}$$

It measures how exploitable the recommended policy $\hat{\pi}_i$ is in comparison to the optimal policy for g.

But suppose that we are uncertain about g. More precisely, suppose the reward and transition functions of g are parameterized by some unknown parameter $\theta : \Theta$, where our beliefs about θ are modelled by some distribution $D_\Theta : \triangle\Theta$ that induces a corresponding distribution $D : \triangle\mathcal{G}$ over games (we call this the *belief distribution*). In that case, to maximize our expected utility, we should recommend

$$\operatorname*{argmax}_{\pi_i} \mathop{\mathbb{E}}_{g\sim D} \min_{\pi_{-i}} u_i(g, \pi) \tag{3}$$

Unfortunately, this typically has no closed-form solution. Instead, we replace the mean with a sample mean:

$$\hat{\pi}_i = \operatorname*{argmax}_{\pi_i} \sum_j \min_{\pi_{-i}} u_i(g_j, \pi) \tag{4}$$

where $g_j \sim D$ for $j \in \mathcal{J}$. That is, we find a policy that optimizes over an *ensemble* of games, rather than a single game. The probability that the optimality gap

$$\max_{\pi_i} \mathop{\mathbb{E}}_{g\sim D} \min_{\pi_{-i}} u_i(g, \pi) - \mathop{\mathbb{E}}_{g\sim D} \min_{\pi_{-i}} u_i(g, \hat{\pi}_i, \pi_{-i}) \tag{5}$$

exceeds $\varepsilon \geq 0$ is at most $e^{-2|\mathcal{J}|\varepsilon^2/\Delta^2}$, where Δ is i's payoff range (maximum possible payoff minus minimum possible payoff). See the appendix for a proof.

Computation

We can compute this policy through backward induction. Let

$$v : \prod_j \prod_s \mathbb{R} \quad q : \prod_j \prod_s \prod_a \mathbb{R} \quad \pi_i : \prod_s \triangle\mathcal{A}_i(s) \tag{6}$$

where v yields a value for each state in each game, q yields a value for each action profile in each state in each game, and π_i is a policy for i. Let r_j and δ_j be the reward and transition functions of g_j. Then let

$$v_j(s) = \min_{a_{-i}} q_j(s, \pi_i(s), a_{-i}) \tag{7}$$

$$q_j(s, \sigma_i, a_{-i}) = \sum_{a_i} \sigma_i(a_i) q_j(s, a) \tag{8}$$

$$q_j(s, a) = r_{j,i}(s, a) + \sum_{s'} \delta_j(s, a, s') v_j(s') \tag{9}$$

$$\pi_i(s) = \operatorname*{argmax}_{\sigma_i} \sum_j \min_{a_{-i}} q_j(s, \sigma_i, a_{-i}) \tag{10}$$

where $v_j(s) \stackrel{\text{def}}{=} 0$ for terminal states. To compute the first and last lines, we formulate and solve the linear program

$$\begin{aligned}
&\text{maximize} \quad \mathbf{1} \cdot \alpha \quad \text{over} \quad \alpha : \mathbb{R}^{\mathcal{J}} \quad \beta : \mathbb{R}^{\mathcal{A}_i(s)} && (11)\\
&\text{subject to} \quad \mathbf{1} \cdot \beta = 1 && (12)\\
&\qquad\qquad \beta \geq \mathbf{0} && (13)\\
&\qquad\qquad \alpha \leq \Gamma(a_{-i}) \cdot \beta \quad \forall a_{-i} : \mathcal{A}_{-i}(s) && (14)
\end{aligned}$$

where $\Gamma(a_{-i})_j = q_j(s, a_{-i})$, and let $v_j(s) = \alpha_j, \pi_i(s) = \beta$. The dual variables associated with the inequality constraints contain best responses from player $-i$, that is, $\pi_{-i,j}(s)$.

The recommended policy is a non-local function of both observations and the initial belief distribution. Depending on the belief distribution, there may be nontrivial correlations across states, including unvisited states. For example, if we believe there is a treasure chest at location A or B (but not both), then once we have explored A and found nothing, we will know that the treasure chest is at B and behave accordingly, despite not having visited B. This example illustrates the advantage of our Bayesian approach, which allows one to incorporate any useful domain-specific knowledge into the initial belief distribution.

Exploration policies for games

Our problem is as follows: We face a stochastic game with unknown rewards and/or transitions. Our beliefs about the true game are modelled by a distribution D over possible games, from which we have the ability to sample. For each of T episodes, we choose a *policy profile*, observe an episode of gameplay under that policy profile—including the transitions and (possibly stochastic) rewards on each timestep—and update our beliefs according to these observations. After T episodes, we recommend a policy for Player i (see Equation 4). This leaves the question of how to explore over the course of the T episodes, since this determines how useful the beliefs we end up with will be. In this section, we design a number of exploration policies, and later in the paper compare them experimentally.

Let $v_t(D)$ denote the expected utility of a belief distribution D when t exploration episodes remain. Then

$$v : \mathbb{N} \to \triangle\mathcal{G} \to \mathbb{R} \tag{15}$$

$$v_0(D) = \max_{\pi_i} \mathop{\mathbb{E}}_{g\sim D} \min_{\pi_{-i}} u_i(g, \pi) \tag{16}$$

$$v_{t+1}(D) = \max_{\pi} \mathop{\mathbb{E}}_{D'|D,\pi} v_t(D') \tag{17}$$

where $D'|D, \pi$ denotes the belief distribution we acquire after an episode, conditioned on the fact that D is our belief distribution before the episode and π is the policy profile we use during the episode.

Thus, if our current belief distribution is D and there are t remaining exploration episodes, the optimal policy profile to use for exploration is $\operatorname{argmax}_\pi \mathbb{E}_{D'|D,\pi}\, v_t(D')$, which can be computed recursively as described above. However, this computation is intractable. The nested maximizations form a tree whose depth is the number of remaining episodes, and the branching factor (the number of policy profiles to optimize over in each maximization) is itself exponential in the size of the game. Even computing the optimal exploration policy with a single-step look-ahead (that is, letting $t = 1$) is intractable, for the latter reason.

In the rest of the paper, we will design exploration policies that are more computationally feasible. Our first policy is the **uniform** policy, which simply selects actions for all players uniformly at random: $\pi_i(s) = \operatorname{uniform}(\mathcal{A}_i(s))$. The **min-count** policy selects in each state the action profile that has been chosen the least so far: $a = \operatorname{argmin}_a \#(s, a)$, where $\#(s, a)$ is the number of times a has been chosen in s. Ties are broken uniformly at random. The **greedy** policy uses the argmax-sum-min policy given by Equation 4 both players.

The main idea behind this policy (as well as other policies described below) is that, if we are reasonably certain about where the Nash equilibrium of the game is, it makes sense to try to refine our knowledge by staying near this equilibrium, rather than waste time on game trajectories that we believe are far away from equilibrium gameplay.

The greedy policy, at least in the limit of infinite samples from the belief distribution, is not guaranteed to converge to correct beliefs about the game, because it may get stuck with false max-mean-min policies for both players from which it continuous to sample forever, ignoring the rest of the game.

To solve this problem, we can force the greedy policy to explore more by combining it with the uniform policy, yielding the ε**-greedy** policy. More precisely, at each individual timestep, ε-greedy chooses to execute the uniform policy instead of the greedy policy with probability ε.

Our next exploration policy is inspired by Thompson sampling (Thompson 1933), a remarkably simple successful heuristic for solving exploration-exploitation dilemmas in multi-armed bandit problems (Ortega and Braun 2010). It consists of playing an action according to the probability that it is the optimal action. This is done by sampling a belief from the belief distribution and then acting optimally with respect to that belief. It has become a popular approach for RL (Russo et al. 2018), and convergence results have been obtained that show it is asymptotically optimal and well behaved (Osband and Roy 2017; Agrawal and Jia 2017).

Our **Thompson policy**, a generalization of Thompson sampling to games, samples a single game from our belief distribution at the beginning of each episode, and then uses the argmax-min policies (Equation 1) for both players in that game for the duration of the episode. Like the greedy exploration policy, it biases gameplay toward what is believed to be the Nash equilibrium, but unlike greedy exploration, it leaves more room for exploration according to how uncertain the agent is. The more certain we are of the true game, and thus of where its Nash equilibrium lies, the more often we will sample trajectories played by that equilibrium, refining our knowledge of that region of the game.

Recall that our ultimate goal is to recommend a policy for player i, assuming $-i$ will play optimally against it. Because of this, we also propose the **semi-Thompson** policy, which uses $-i$'s *best response* to i's argmax-min policy, rather than $-i$'s own argmax-min policy.

Auer, Cesa-Bianchi, and Fischer (2002) introduced the UCB1 algorithm for multi-armed bandit problems, proving that it achieves optimal regret up to a multiplicative constant. UCB1 selects the action a with the highest *upper confidence bound* (UCB) on its utility, where (for utilities in $[0, 1]$),

$$\operatorname*{UCB1}_{g\sim D} f(g, a) = \operatorname*{\mathbb{E}}_{g\sim D} f(g, a) + \sqrt{\frac{2 \ln \sum_{a'} \#(a')}{\#(a)}} \quad (18)$$

This type of approach is known as "optimism in the face of uncertainty", and it encourages exploration of actions with more uncertain payoffs. In theory, the analogous policy for our setting would be

$$\operatorname*{argmax}_{\pi_i} \operatorname*{UCB1}_{g\sim D} \min_{\pi_{-i}} u_i(g, \pi) \quad (19)$$

Unfortunately, we do not have a tractable way to compute this π_i. Instead, we use the following approximation:

First, we compute the value $Q_i(g, s, a)$ of each state and action profile for each *individual* game g under Nash equilibrium gameplay *in that game*. To do this, we reuse the backward induction algorithm but let $\mathcal{J} = \{j\}$ be a singleton, where g_j is the individual game under consideration.

Having done this, our **UCB1** policy is

$$\pi_i(s) = \operatorname*{argmax}_{\sigma_i} \operatorname*{UCB1}_{g\sim D} \min_{\sigma_{-i}} Q_i(g, s, \sigma), \text{ where} \quad (20)$$

$$\operatorname*{UCB1}_{g\sim D} f(g, \sigma_i) = \operatorname*{\mathbb{E}}_{g\sim D} f(g, \sigma_i) + \sqrt{\frac{2 \ln \sum_{a_i} \#(a_i)}{\#(\sigma_i(a_i))}} \quad (21)$$

where we define an analog of action counts for mixed strategies, consisting of their expected action count:

$$\#(\sigma_i(a_i)) = \sum_{a_i} \sigma_i(a_i) \#(a_i) \quad (22)$$

We approximate the mean with an *empirical* mean over the samples from our belief distribution, $\{g_j\}_{j:\mathcal{J}}$.

$$\operatorname*{\mathbb{E}}_{g\sim D} f(g, \sigma_i) \approx \frac{1}{|\mathcal{J}|} \sum_j f(g_j, \sigma_i) \quad (23)$$

Kaufmann, Cappe, and Garivier (2012) introduced a Bayesian version of UCB and proved that it satisfies finite-time regret bounds that imply asymptotic optimality. It selects the action whose mean payoff $1 - \frac{1}{t}$ quantile is highest, where t is the current exploration episode. Our **Bayes-UCB** policy is the same as the UCB1 policy, except that we modify the UCB:

$$\operatorname*{UCB1}_{g\sim D} f(g, \sigma_i) = \operatorname*{quantile}_{g\sim D} f(g, \sigma_i) \quad (24)$$

We approximate the quantile with a linearly-interpolated *empirical* quantile over samples from our belief distribution.

We then observed that, in a zero-sum game, it is not possible to be optimistic about payoffs for both players simultaneously in an internally self-consistent way—because if the payoff is high for one, it must be low for the other. Thus, for zero-sum games, UCB-like policies like the above are conceptually questionable, because the optimistic assumptions they make seem to be inconsistent. Therefore, we introduce an exploration policy that we call *exploitability optimism*. It is motivated by the success of optimism in exploration in single-agent settings, but does not use the inconsistent optimism about the players' payoffs. Rather, it samples an action profile from the strategy profile that has the lowest lower confidence bound (LCB) on the players' combined *exploitabilities*. More precisely, our **sum-exploit** and **max-exploit** policies are

$$\pi(s) = \operatorname*{argmin}_{\sigma} \operatorname*{LCB}_{g \sim D} f(\text{expl}(Q(g, s), \sigma)) \tag{25}$$

for $f(x) = \sum_i x_i$ and $f(x) = \max_i x_i$ respectively, where

$$\text{expl}(u, \sigma)_i = \max_{\sigma_i'} u_i(\sigma_i', \sigma_{-i}) - u_i(\sigma) \tag{26}$$

measures the suboptimality of player i's strategy under payoffs u and strategy profile σ. As before, we approximate the LCB with a linearly-interpolated empirical quantile—this time of $\frac{1}{t}$, where t is the current exploration episode–over the samples from our belief distribution, $\{g_j\}_{j:\mathcal{J}}$.

Strategy optimization. For the exploration policies that require optimization over a space of strategies or strategy profiles and lack a closed-form solution—namely UCB1, Bayes-UCB, sum-exploit, and max-exploit—we replace optimization over the uncountable space $\triangle\mathcal{A}_i(s)$ with optimization over a finite subset $\mathcal{X}_i \subseteq \triangle\mathcal{A}_i(s)$ of strategies sampled uniformly at random from $\triangle\mathcal{A}_i(s)$.

Convergence. In our experiments, presented later, all these exploration policies appeared to converge to zero regret. Even the greedy exploration policy did not get stuck because we use a stochastic estimator based on samples from D. Convergence is *guaranteed* if throughout the exploration process, for each action profile, there is a nonzero probability of sampling a game g such that the algorithm-selected Nash equilibrium of g plays that action profile with nonzero probability. This, in turn, can be *guaranteed* by adding a small amount of uniform randomness to any of the exploration policies, as we did with ε-greedy. (In our experiments, we did not do that, because they converged anyway.)

The key question is how quickly these exploration policies converge to low-exploitability strategies. Although establishing theoretical worst-case bounds of convergence for our methods would be interesting future work, methods with the best theoretical bounds are not necessarily those that perform best in practice—as the single-agent reinforcement learning literature shows. For that reason, we consider it important to investigate how quickly our methods converge *in practice*.

Experiments

We compared the performance of the exploration policies we proposed. We measure how quickly the regret of the recommended policy decreases as a function of episodes in the learning phase. We experiment in small-to-medium-sized games in order to be able to evaluate the regret exactly.

We use two kinds of benchmarks. The first kind consists of randomly-generated normal-form games with binary $\{0, 1\}$ (that is, Bernoulli-distributed) rewards. We assume the agent does not know the mean reward (that is, the Bernoulli parameter) associated with any action profile. The agent starts with uniform priors (which are a special case of beta priors) over the parameter for each action profile. Because the beta distribution is a conjugate prior for the Bernoulli distribution, we can easily update these priors according to incoming observations—by simply incrementing the pseudocount corresponding to the observed reward.

Figures 1 and 4 illustrates the performance of different exploration policies on such 10×10 normal-form games. We used 100 trials. For the policies requiring it, 100 game samples and 100 strategy or strategy profile samples were used. For the ε-greedy exploration policy we used $\varepsilon = 0.1$. Our Python 3.8 implementation used Gurobi 9.1 and ran on a 2.5 GHz Quad-Core Intel Core i7 processor.

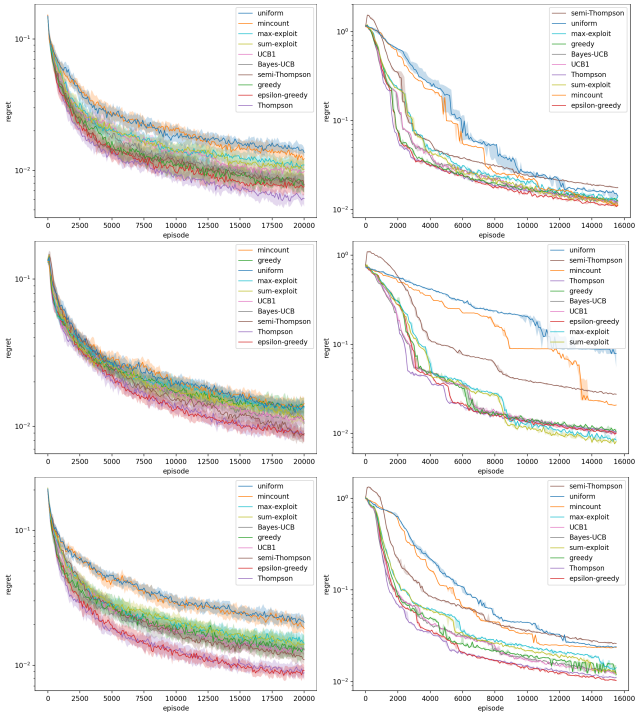

Figure 1: Exploration policies on normal-form games (left) and bombardment games (right). See apppendix for more figures.

As expected, the naive exploration policies—uniform and mincount—underperform the more sophisticated exploration policies. The ε-greedy policy attains better performance than the greedy policy. The UCB1, Bayes-UCB, and exploitability policies also perform well. Our Thompson and semi-Thompson policies are consistently among the best-performing policies. Thompson has an advantage in terms of computational efficiency (see Figure 2) because it only needs to sample and solve *one* game from its belief distribu-

tion.

One can generalize this to multi-state stochastic games with unknown transition probabilities by using a Dirichlet distribution (which is the conjugate prior of the categorical distribution) for each state and action profile, incrementing the pseudocount corresponding to the observed next state. Our second kind of benchmark studies stochastic games where the rewards are known but the transition probabilities are not. To study this setting, we created a benchmark called the *bombardment game*, illustrated in Figure 3. It is a gridworld-like setting which starts with Player 1 (the solid disk) in the top-left corner and ends when they reach the bottom-right corner. In each time step, Player 1 stays put or chooses one of the 4 cardinal directions to move in. Player 2 (the ring) simultaneously chooses to bomb either Player 1's current position or one of its 4 neighboring positions. Player 1's objective is to minimize their damage. They accomplishes this by reaching the goal as quickly as possible while remaining unpredictable enough to dodge Player 2's bombardments to an optimal extent. Player 1 receives a reward of -1 whenever their next position coincides with Player 2's ring. The layouts we tested are shown in Figure 3. Each episode had a 40-step horizon.

Player 1 learned to prefer areas with more room for maneuvering and proximity to the goal. Player 2 knows this and learned its bombardment policy accordingly. This interplay resulted in complex emergent behavior. The performance of the different exploration policies on the gap, nest, and double fork layouts is shown in Figure 5.

Compared to the normal-form game setting, we observe some significant differences in the performance of the exploration policies. One important difference is that the semi-Thompson policy consistently performs worse than almost all the other policies, and asymptotically seems to be outpaced by them. This might be due to the fact that it does not perform enough exploration because it uses only the deterministic best response for Player 2 to Player 1's policy (under the sampled game). Curiously, its regret increases sharply in the initial phase of exploration. Another important difference is that there is a much larger gap between the naive exploration policies (uniform and mincount) and the other exploration policies than in the normal-form game setting, at least initially. This suggests that, in larger and more complex environments, there is a bigger advantage in adopting more sophisticated exploration policies. Finally, our Thompson policy is still consistently among the best-performing policies, but the exploitability policies tend to outperform the rest more frequently than they do in the normal-form game setting. For all of the exploration policies except mincount, uniform, greedy, and ε-greedy, the dynamic program across the states is the same. The computation time scales linearly with the number of states and with the length of the horizon. The relative run time of the different policies is thus the same as in the normal-form case shown in Figure 2. Therefore, our Thompson policy can be recommended due to both its solution quality and its speed advantage.

It is not necessary to start with a uniform prior. For example, we may be certain there are only two possible environments, in which case the agent's beliefs are modeled by a mixture of the two. Our exploration policies are independent of the particular form or implementation of the belief distribution, since they only require the ability to sample from it.

We also compare our methods to the model-free minimax Q-learning algorithm (Littman 1994). The results, in the appendix, show that our model-based methods are one or two orders of magnitude more sample efficient!

Conclusions and future research

We investigated the increasingly important and common game-solving setting where we do not have an explicit description of the game but only oracle access to it through game play. During a limited-duration learning phase, the algorithm can control the actions of both players in order to try to learn the game and how to play it well. After that, the algorithm has to produce a policy that has low exploitability. We generalized exploration policies that are used in the single-agent settings to games. We showed that games raise additional issues and some generalizations are inherently inconsistent, and we introduced new ones that fix that. We proposed using the distribution of state-action value functions induced by a belief distribution over possible environments.

We conducted experiments on normal-form and stochastic games where exploitability can be evaluated exactly. The more sophisticated exploration policies—particularly our Thompson sampling-based policy, which is also computationally efficient—tended to outperform the more naive exploration policies in solution quality. We also showed one to two orders of magnitude sample-efficiency improvement over a well-known model-free exploration approach.

There are several ways to extend this work. First, one could replace exact game and game-ensemble solving with approximate solving, speeding up the computation time of the exploration algorithms that rely on these solutions.

Second, one could extend this work to imperfect-information extensive-form games. For example, one could use the *sequence form* (von Stengel 1996) representation and adapt Equation 4 to work over that setting, which is still a linear program. Alternatively, one could work directly with the tree form by creating a root chance node whose children are the games in the ensemble, distinguishing the information sets (across the ensemble) that belong to player $-i$ but not player i, and then running a standard game-solving algorithm such as CFR (Zinkevich et al. 2007) or its modern variants (Brown and Sandholm 2015, 2017, 2019; Brown, Kroer, and Sandholm 2017; Farina, Kroer, and Sandholm 2021). This will retrieve the argmax-sum-min policy for i. Since those algorithms are iterative, one can get fast approximate solutions (as mentioned earlier) by terminating the process early, at the start of each exploration episode. Solutions from previous episodes could also be used to warm-start the process (Brown and Sandholm 2016).

Finally, one could extend this work to environments with large and/or continuous state spaces where function approximation is required. Our exploration policies suggest maintaining an ensemble of neural networks modelling the agent's belief distribution over environments. Each network models the transition and reward functions of a hallucinated

environment, and is repeatedly trained on observation tuples accumulated over the course of exploration. In parallel, one trains separate Q-value networks on the inputs and outputs of each of these environment networks. This yields an ensemble of Q functions (one for each environment in the belief distribution) that can be used by our exploration algorithms.

Acknowledgements

This material is based on work supported by the National Science Foundation under grants IIS-1718457, IIS-1901403, and CCF-1733556, and the ARO under award W911NF2010081.

References

Agrawal, S.; and Jia, R. 2017. Optimistic posterior sampling for reinforcement learning: worst-case regret bounds. In *NIPS*.

Auer, P.; Cesa-Bianchi, N.; and Fischer, P. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine learning* .

Bellemare, M.; Dabney, W.; and Munos, R. 2017. A distributional perspective on reinforcement learning. In *ICML*.

Berner, C.; Brockman, G.; Chan, B.; Cheung, V.; Debiak, P.; Dennison, C.; Farhi, D.; Fischer, Q.; Hashme, S.; Hesse, C.; Józefowicz, R.; Gray, S.; Olsson, C.; Pachocki, J.; Petrov, M.; de Oliveira Pinto, H. P.; Raiman, J.; Salimans, T.; Schlatter, J.; Schneider, J.; Sidor, S.; Sutskever, I.; Tang, J.; Wolski, F.; and Zhang, S. 2019. Dota 2 with Large Scale Deep Reinforcement Learning. *CoRR* URL http://arxiv.org/abs/1912.06680.

Brown, N.; Kroer, C.; and Sandholm, T. 2017. Dynamic Thresholding and Pruning for Regret Minimization. In *AAAI*.

Brown, N.; and Sandholm, T. 2015. Regret-Based Pruning in Extensive-Form Games. In *NIPS*.

Brown, N.; and Sandholm, T. 2016. Strategy-Based Warm Starting for Regret Minimization in Games. In *AAAI*.

Brown, N.; and Sandholm, T. 2017. Reduced Space and Faster Convergence in Imperfect-Information Games via Pruning. In *ICML*.

Brown, N.; and Sandholm, T. 2019. Solving imperfect-information games via discounted regret minimization. In *AAAI*.

Bubeck, S.; Munos, R.; and Stoltz, G. 2009. Pure Exploration in Multi-armed Bandits Problems. In *ALT*.

Casgrain, P.; Ning, B.; and Jaimungal, S. 2019. Deep Q-Learning for Nash Equilibria: Nash-DQN. *CoRR* URL http://arxiv.org/abs/1904.10554.

Chen, R.; Sidor, S.; Abbeel, P.; and Schulman, J. 2017. UCB and InfoGain Exploration via Q-Ensembles. *CoRR* URL http://arxiv.org/abs/1706.01502.

Claus, C.; and Boutilier, C. 1998. The dynamics of reinforcement learning in cooperative multiagent systems. In *AAAI*.

Conitzer, V.; and Sandholm, T. 2003. BL-WoLF: A Framework For Loss-Bounded Learnability In Zero-Sum Games. In *ICML*.

Dearden, R.; Friedman, N.; and Russell, S. 1998. Bayesian Q-learning. In *AAAI*.

Farina, G.; Kroer, C.; and Sandholm, T. 2021. Faster Game Solving via Predictive Blackwell Approachability: Connecting Regret Matching and Mirror Descent. In *AAAI*.

Ganzfried, S.; and Sandholm, T. 2009. Computing equilibria in multiplayer stochastic games of imperfect information. In *IJCAI*.

Garivier, A.; Kaufmann, E.; and Koolen, W. M. 2016. Maximin Action Identification: A New Bandit Framework for Games. In *JMLR*.

Heinrich, J.; and Silver, D. 2016. Deep Reinforcement Learning from Self-Play in Imperfect-Information Games. *CoRR* URL http://arxiv.org/abs/1603.01121.

Hu, J.; and Wellman, M. 2003. Nash Q-Learning for general-sum stochastic games. *JMLR* .

Kaufmann, E.; Cappe, O.; and Garivier, A. 2012. On Bayesian upper confidence bounds for bandit problems. In *AISTATS*.

Lanctot, M.; Zambaldi, V.; Gruslys, A.; Lazaridou, A.; Tuyls, K.; Perolat, J.; Silver, D.; and Graepel, T. 2017. A Unified Game-Theoretic Approach to Multiagent Reinforcement Learning. In *NIPS*.

Littman, M. 1994. Markov games as a framework for multi-agent reinforcement learning. In *ICML*.

Lockhart, E.; Lanctot, M.; Pérolat, J.; Lespiau, J.-B.; Morrill, D.; Timbers, F.; and Tuyls, K. 2019. Computing Approximate Equilibria in Sequential Adversarial Games by Exploitability Descent. In *IJCAI*.

Marchesi, A.; Trovò, F.; and Gatti, N. 2019. Learning Probably Approximately Correct Maximin Strategies in Simulation-Based Games with Infinite Strategy Spaces. In *AAMAS*.

Mavrin, B.; Yao, H.; Kong, L.; Wu, K.; and Yu, Y. 2019. Distributional reinforcement learning for efficient exploration. In *ICML*.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; Petersen, S.; Beattie, C.; Sadik, A.; Antonoglou, I.; King, H.; Kumaran, D.; Wierstra, D.; Legg, S.; and Hassabis, D. 2015. Human-level control through deep reinforcement learning. *Nature* .

O'Donoghue, B.; Osband, I.; Munos, R.; and Mnih, V. 2018. The uncertainty Bellman equation and exploration. In *ICML*.

Ortega, P. A.; and Braun, D. A. 2010. A minimum relative entropy principle for learning and acting. *JAIR* .

Osband, I.; Blundell, C.; Pritzel, A.; and Roy, B. V. 2016. Deep exploration via bootstrapped DQN. In *NIPS*.

Osband, I.; and Roy, B. V. 2017. Why is Posterior Sampling Better than Optimism for Reinforcement Learning? In *PMLR*.

Osband, I.; Roy, B. V.; Russo, D. J.; and Wen, Z. 2019. Deep exploration via randomized value functions. *JMLR* .

Russo, D.; van Roy, B.; Kazerouni, A.; Osband, I.; and Wen, Z. 2018. A Tutorial on Thompson Sampling. *Foundations and Trends in Machine Learning* .

Sandholm, T.; and Crites, R. 1996. Multiagent Reinforcement Learning in the Iterated Prisoner's Dilemma. *Biosystems* .

Shapley, L. 1953. Stochastic Games. *PNAS* .

Sokota, S.; Ho, C.; and Wiedenbeck, B. 2019. Learning deviation payoffs in simulation-based games. In *AAAI*.

Srinivasan, S.; Lanctot, M.; Zambaldi, V.; Perolat, J.; Tuyls, K.; Munos, R.; and Bowling, M. 2018. Actor-Critic Policy Optimization in Partially Observable Multiagent Environments. In *NeurIPS*.

Thompson, W. 1933. On the Likelihood that One Unknown Probability Exceeds Another in View of the Evidence of Two Samples. *Biometrika* .

Vinyals, O.; Babuschkin, I.; Czarnecki, W. M.; Mathieu, M.; Dudzik, A.; Chung, J.; Choi, D. H.; Powell, R.; Ewalds, T.; Georgiev, P.; Oh, J.; Horgan, D.; Kroiss, M.; Danihelka, I.; Huang, A.; Sifre, L.; Cai, T.; Agapiou, J. P.; Jaderberg, M.; Vezhnevets, A. S.; Leblond, R.; Pohlen, T.; Dalibard, V.; Budden, D.; Sulsky, Y.; Molloy, J.; Paine, T. L.; Gulcehre, C.; Wang, Z.; Pfaff, T.; Wu, Y.; Ring, R.; Yogatama, D.; Wünsch, D.; McKinney, K.; Smith, O.; Schaul, T.; Lillicrap, T.; Kavukcuoglu, K.; Hassabis, D.; Apps, C.; and Silver, D. 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature* .

von Stengel, B. 1996. Efficient Computation of Behavior Strategies. *Games and Economic Behavior* .

Vorobeychik, Y.; and Wellman, M. 2009. Strategic analysis with simulation-based games. In *WSC*.

Wang, X.; and Sandholm, T. 2002. Reinforcement Learning to Play An Optimal Nash Equilibrium in Team Markov Games. In *NIPS*.

Watkins, C.; and Dayan, P. 1992. Q-Learning. *Machine Learning* .

Zhou, Y.; Li, J.; and Zhu, J. 2020. Posterior sampling for multi-agent reinforcement learning: solving extensive games with imperfect information. In *ICLR*.

Zinkevich, M.; Bowling, M.; Johanson, M.; and Piccione, C. 2007. Regret Minimization in Games with Incomplete Information. In *NIPS*.

Appendix

In this appendix we present additional technical material that did not fit in the body of the paper.

Upper bound on the suboptimality of the argmax-sum-min estimator

Theorem 1. Let $f : X \times Y \to \mathbb{R}$. Let D be a probability distribution on Y. Let

$$\hat{x} \in \operatorname*{argmax}_x \frac{1}{n} \sum_i f(x, y_i) \tag{27}$$

where $i \in [n], y_i \sim D$. Let $y \sim D$. The probability

$$\max_x \mathop{\mathbb{E}}_y f(x, y) - \mathop{\mathbb{E}}_y f(\hat{x}, y) \tag{28}$$

exceeds ε is at most $e^{-2n\varepsilon^2/\Delta^2}$, where Δ is the range (supremum minus infimum) of f.

Proof. By Hoeffding's inequality,

$$p = \Pr_{y_i}\left(\mathop{\mathbb{E}}_y f(x, y) - \mathop{\mathbb{E}}_y f(\hat{x}, y) \geq \varepsilon\right) \tag{29}$$

$$= \Pr_{y_i}\left(\mathop{\mathbb{E}}_y f(x, y) - \mathop{\mathbb{E}}_y f(\hat{x}, y) + \frac{1}{n}\sum_i f(x, y_i)\right. \tag{30}$$

$$\left. -\frac{1}{n}\sum_i f(x, y_i) \geq \varepsilon\right) \tag{31}$$

$$\leq \Pr_{y_i}\left(\mathop{\mathbb{E}}_y f(x, y) - \mathop{\mathbb{E}}_y f(\hat{x}, y) + \frac{1}{n}\sum_i f(\hat{x}, y_i)\right. \tag{32}$$

$$\left. -\frac{1}{n}\sum_i f(x, y_i) \geq \varepsilon\right) \tag{33}$$

$$= \Pr_{y_i}\left(\mathop{\mathbb{E}}_y (f(x, y) - f(\hat{x}, y))\right. \tag{34}$$

$$\left. -\frac{1}{n}\sum_i (f(x, y_i) - f(\hat{x}, y_i)) \geq \varepsilon\right) \tag{35}$$

$$\leq \exp\left(-\frac{2n\varepsilon^2}{\Delta^2}\right) \tag{36}$$

□

Additional figures

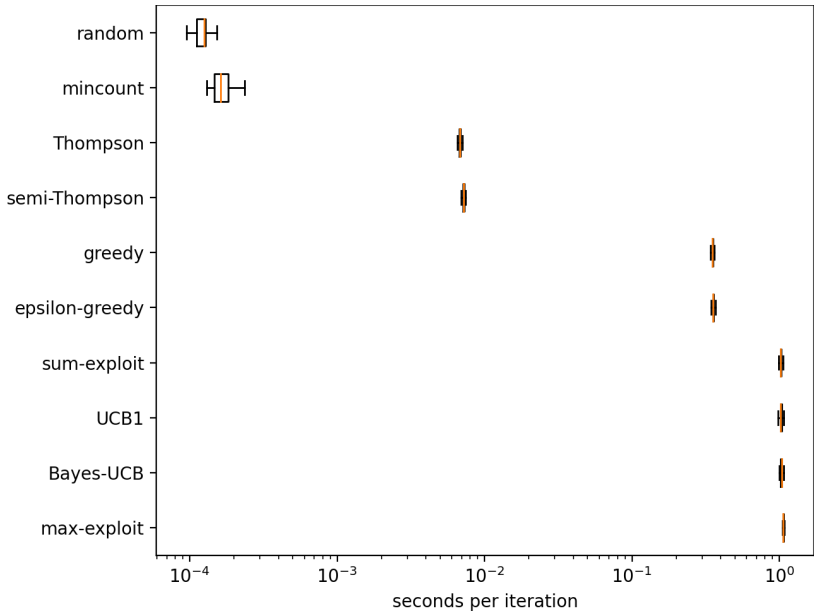

Figure 2: Typical time to compute each exploration policy at the beginning of each exploration episode on a randomly-generated 10×10 normal-form game.

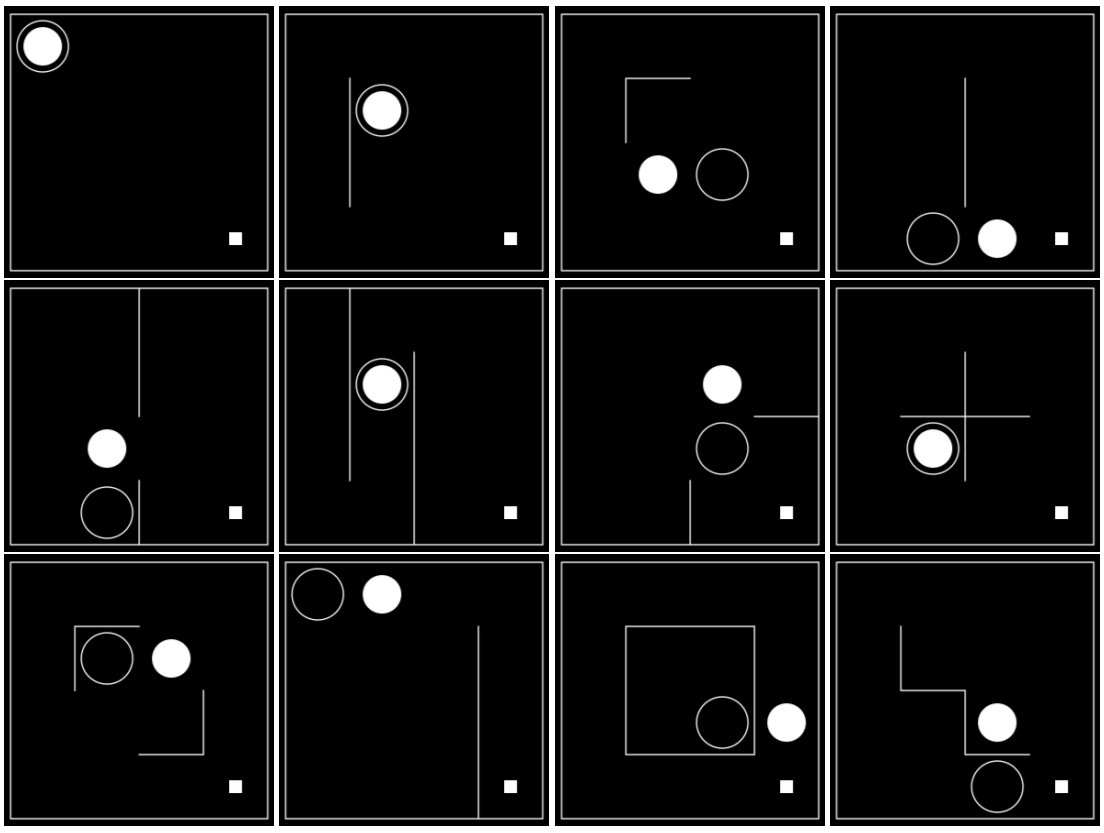

Figure 3: Layouts used for the bombardment environment. Left-to-right, top-to-bottom: empty, corridor, fork, wall, gap, snake, nest, cross, double fork, strip, box, stair.

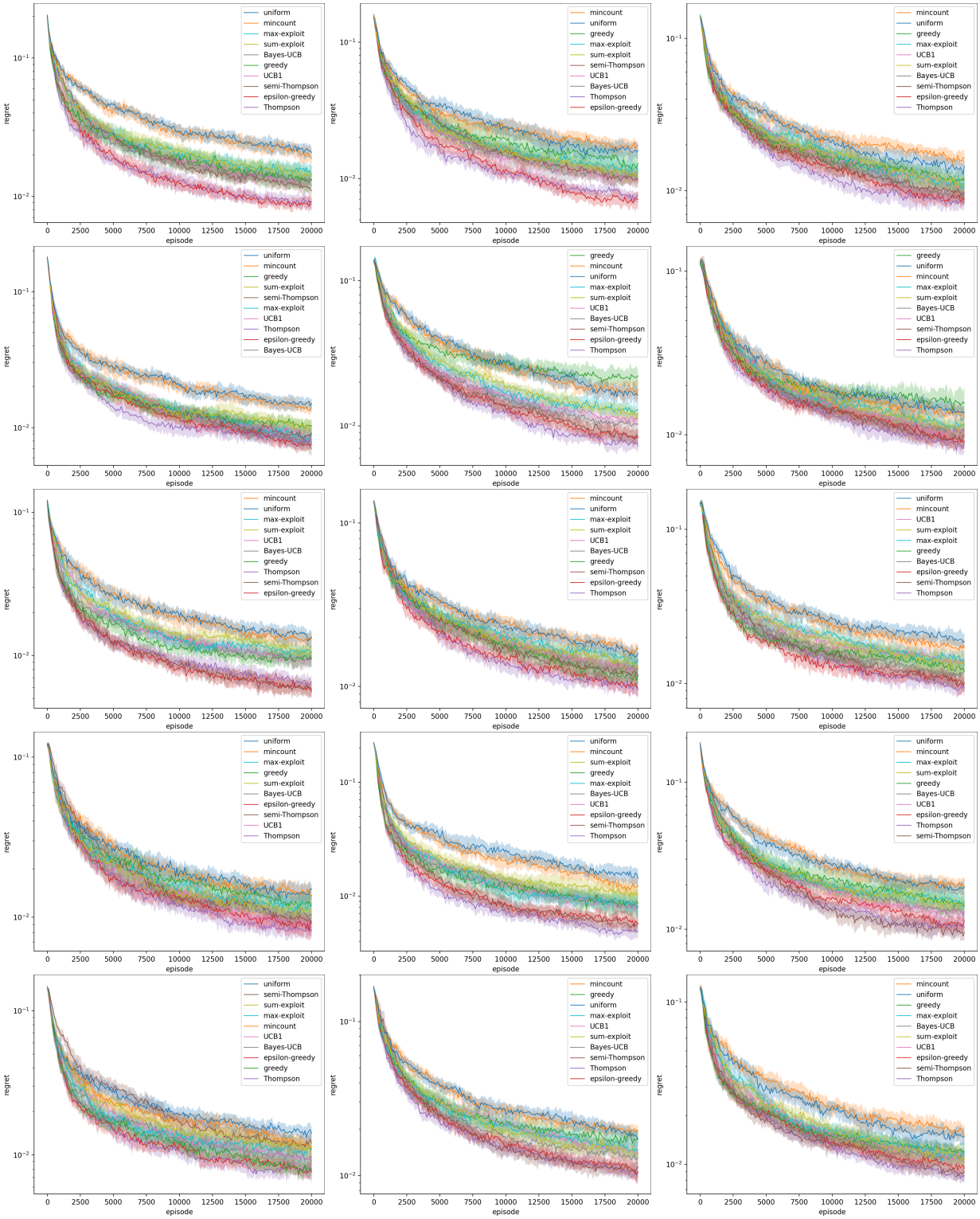

Figure 4: Performance of each policy on more randomly-generated 10×10 normal-form games. Rewards are Bernoulli random variables whose bias is sampled from the uniform distribution, which is also the prior used by the exploratory agent. 100 trials were used. For the policies requiring it, 100 game samples and 100 strategy or strategy profile samples were used on each iteration. For the ε-greedy we used $\varepsilon = 0.1$. The solid curves show the medians and the shaded regions show the 40-60 percentile bands.

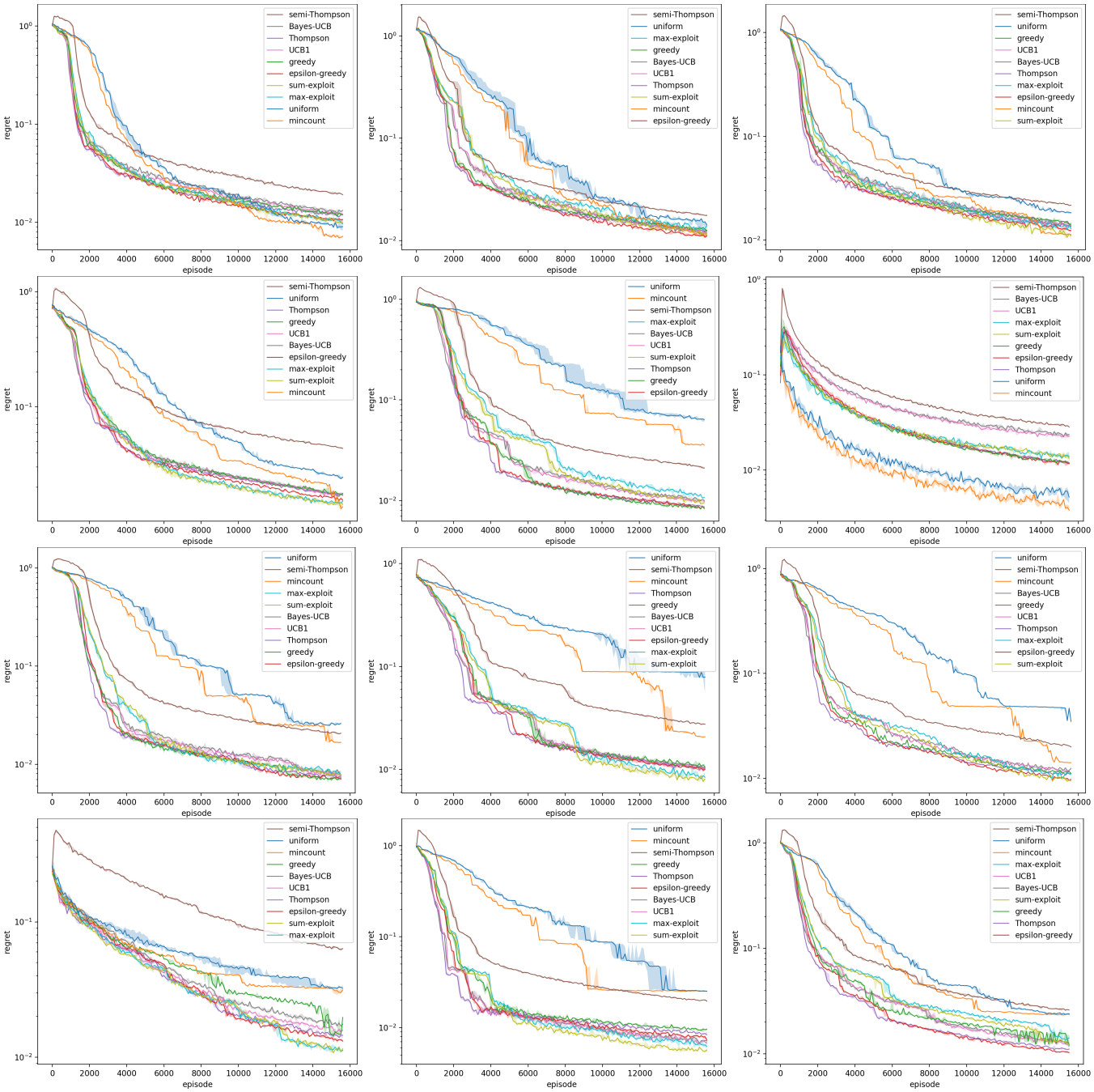

Figure 5: Performance of each policy on each bombardment game layout. Left-to-right, top-to-bottom: empty, corridor, fork, wall, gap, snake, nest, cross, double fork, strip, box, and stair. Each episode had a 40-step horizon. The solid curves show the medians and the shaded regions show the 25-75 percentile bands. Other parameters are the same as for the normal-form game setting.

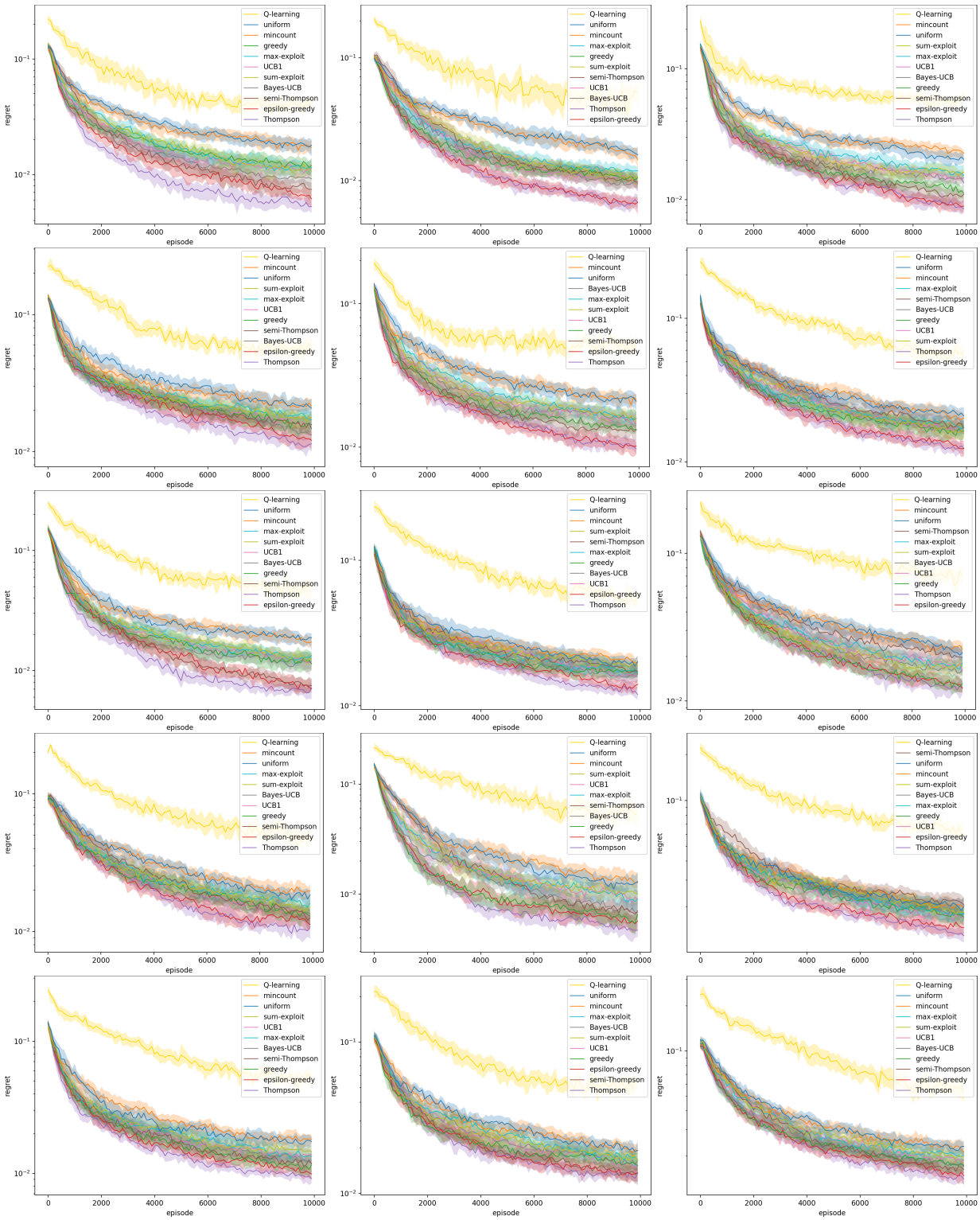

Figure 6: Performance of minimax Q-learning with learning rate 0.1 and ε-greedy exploration ($\varepsilon = 0.1$) on randomly-generated normal form games. Other parameters are the same as for Figure 4.

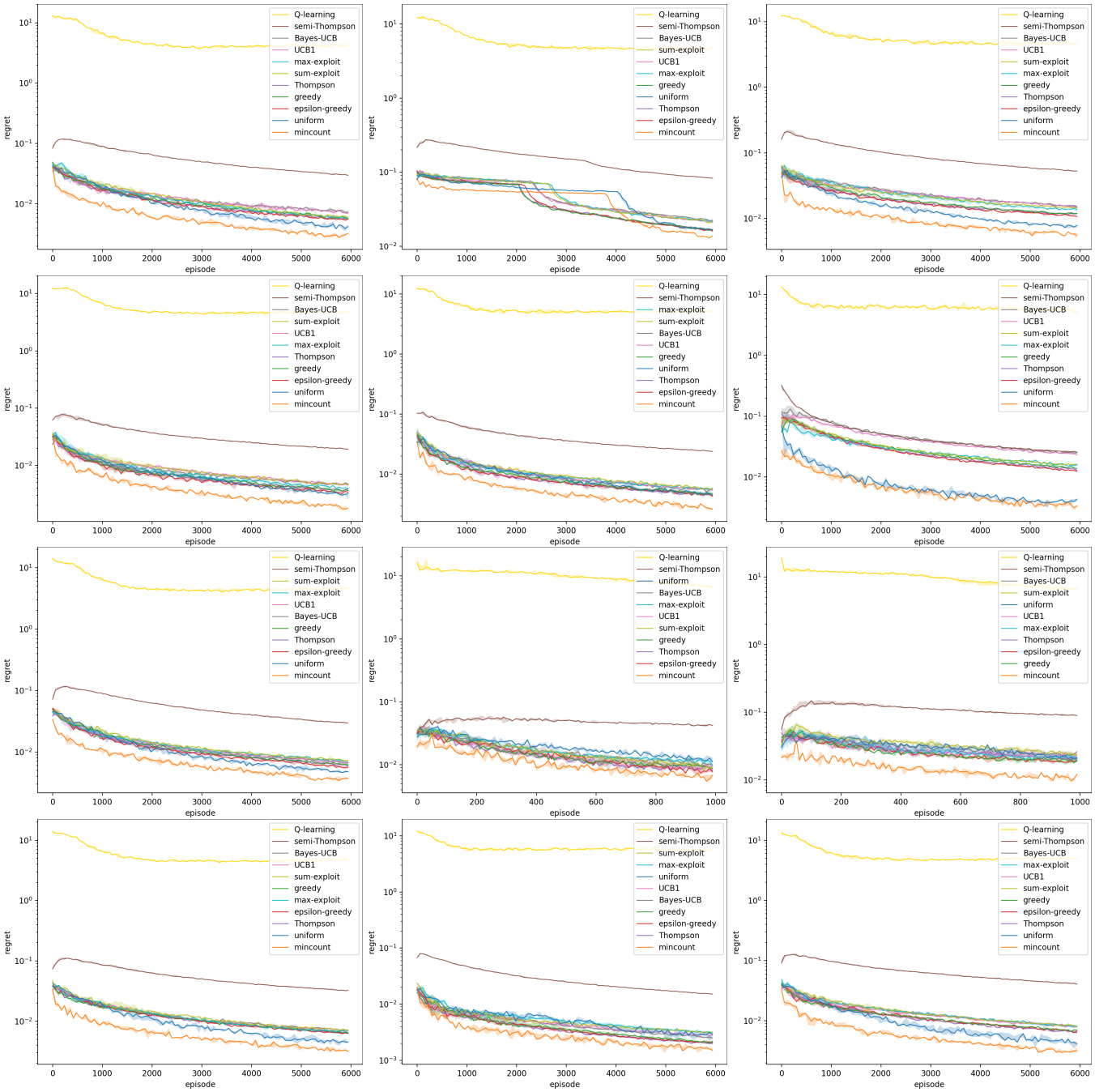

Figure 7: Performance of minimax Q-learning with learning rate 0.1 and ε-greedy exploration ($\varepsilon = 0.1$) on each bombardment game. Other parameters are the same as for Figure 5.