Exploration Analysis in Finite-Horizon Turn-based Stochastic Games

Jialian Li[†], **Yichi Zhou**[†], **Tongzheng Ren**[‡], **Jun Zhu**[†*]
[†]Dept. of Comp. Sci. & Tech., BNRist Center, Institute for AI, Tsinghua-Bosch ML Joint Center, Tsinghua University
[‡]Computer Science Dept., Austin University

Abstract

Exploration and exploitation trade-off is one of the key concerns in reinforcement learning. Previous work on one-player Markov Decision Processes has reached near-optimal results for both PAC and high probability regret guarantees. However, such an analysis is lacking for the more complex stochastic games with multi-players, where all players aim to find an approximate Nash Equilibrium. In this work, we address the exploration issue for the N-player finite-horizon turn-based stochastic games (FTSG). We propose a framework, *Upper Bounding the Values for Players* (UBVP), to guide exploration in FTSGs. UBVP leverages the key insight that players choose the optimal policy conditioning on the policies of the others simultaneously; thus players can explore *in the face of uncertainty* and get close to the Nash Equilibrium. Based on UBVP, we present two provable algorithms. One is *Uniform*-PAC with a sample complexity of $\tilde{O}(1/\epsilon^2)$ to get an ϵ-Nash Equilibrium for arbitrary $\epsilon > 0$, and the other has a cumulative exploitability of $\tilde{O}(\sqrt{T})$ with high probability.

1 INSTRUCTION

Sequential decision-making processes among multi-players are common in practice, such as board games [Silver et al., 2017a] and computer games [Rouse III, 2010]. When the dynamics and rewards of the decision process (i.e., the environment) are known, game theoretical methods can be applied to find a Nash Equilibrium (NE) [Nisan et al., 2007], at which no player is willing to change its current policy individually. When the environment is unknown to players, as in the Multi-agent Reinforcement Learning (MARL) setting [Zhang et al., 2019], finding the NE requires players to exploit their best policies as much as possible while ensuring enough exploration to avoid being trapped in sub-optimal policies. At the same time, players need to take their mutual influence into consideration and finally converge to some approximate NE [Nisan et al., 2007]. Thus, the exploration-exploitation trade-off among all players is an essential issue for MARL problems.

For Markov Decision Process (MDP) that models the interaction between a single player and the environment, substantial progress has been made on solving the exploration-exploitation trade-off. For instance, Azar et al. [2017] reach near optimal regrets of order $\tilde{O}(\sqrt{T})$ for T time steps, while Dann and Brunskill [2015] follow the Probably Approximately Correct (PAC) framework to provide a sample complexity of $\tilde{O}(1/\epsilon^2)$ for an ϵ-optimal policy[1]. Such results handle the exploration-exploitation trade-off with the *Optimism in the Face of Uncertainty* (OFU) principle, i.e. choosing the policy that is optimal under the current uncertainty estimation.

However, for MARL, it is non-trivial to solve the exploration issue. The main difficulty is that the dynamics for each player is no longer static — rewards of one player would change when the policies of other players change. Therefore, we cannot directly extend the techniques in MDP for MARL. Previous efforts has been devoted to solving the stochastic games and extensive games, the two main frameworks for MARL (see [Zhang et al., 2019] for detailed comparison). Methods considering general stochastic games are usually hard to solve and lack a non-asymptotic analysis for exploration [Littman, 1994, Hu and Wellman, 2003]. Many methods concen-

[*]corresponding author (dcszj@tsinghua.edu.cn)

[1]We use $\tilde{O}$ to denote the order ignoring poly-logarithmic terms. Here we ignore other parameters and we leave detailed analysis to latter sections.

trate on two-player (zero-sum) stochastic games and apply techniques from MDP [Wei et al., 2017]. However, they are not suitable to extend to the more challenging N-player cases. There are also some policy-gradient-based methods [Lockhart et al., 2019], which mostly lack provable and efficient exploration. For extensive games, methods like Fictitious self-play (FSP) [Heinrich et al., 2015] and Monte Carlo counterfactual regret minimization with outcome sampling (MCCFROS) [Lanctot et al., 2009] also suffer from the problem of inefficient exploration. Only a recent variant gives a provable solution for two-player zero-sum extensive games under the Bayesian setting [Zhou et al., 2020].

In this work, we focus on the exploration-exploitation trade-off in finding the NEs for *Finite-horizon Turn-based Stochastic Games* (FTSG) with N players. Here, finite-horizon denotes a finite time steps H for one game and turn-based refers to that there is only one player taking an action at each time step. FTSG includes many traditional games like Go [Silver et al., 2017b] and computer games like Civilization series [Rouse III, 2010].

To our best knowledge, there is little work for solving the exploration issue in general FTSGs. Our work follows the *centralized-learning-decentralized-execution* paradigm to give a solution for learning NEs of FTSGs. This paradigm assumes a centralized controller for training and is adopted by many MARL methods [Zhang et al., 2019]. We define two performance measurements for FTSGs, based on the concept of approximate NEs. Then we propose our framework, *Upper Bounding the Values for Players* (UBVP), to identify NEs for FTSGs. UBVP applies the OFU principle in a way that all players are optimal conditioning on the others. Thus they can converge to the best responses of each other. We give a non-asymptotic analysis and show that UBVP can indeed efficiently explore the unknown environment. We further show that UBVP in fact converges to a subgame perfect equilibrium (SPE), which is an NE for all subgames.

Based on UBVP, we present two concrete algorithms — The first one is Uniform-PAC [Dann et al., 2017] and the second one has a high probability exploitability bound. Both algorithms have comparable theoretical results to the state-of-the-art works for single-player MDPs. Finally, to demonstrate the effectiveness of our method, we choose state-of-the-art exploration methods for stochastic games and extensive games as our baselines. Empirical results show that our method performs well. Specifically, on our designed cooperative game where the SPE is the optimal solution, our method can approach the SPE while other methods are trapped in sub-optimal NEs.

2 RELATED WORK

Our work relates to various topics, as reviewed below.

Markov Decision Process: Much work has been done on the exploration and exploitation trade-off in finite-horizon MDPs. There are mainly two kinds of performance measurements. The first is *regret*, which measures the culminated reward difference between the optimal policy and the algorithm's policies. UCBVI [Azar et al., 2017] reaches the near optimal regret of $\tilde{O}(H\sqrt{SAT})$.[2] The other one is *sample complexity* under the Probably Approximately Correct (PAC) framework to measure the number of time steps needed for an approximately optimal policy. UCFH [Dann and Brunskill, 2015] has a sample complexity upper bound of $\tilde{O}(H^3S^2A/\epsilon^2)$. Moreover, Dann et al. [2017] propose a Uniform-PAC framework to get a PAC solution without ϵ being given.

If the immediate reward is a d-dimensional vector, an MDP turns to be a Multi-Objective MDP (MOMDP) [Roijers et al., 2013]. An MOMDP only involves one agent, which just considers how to trade-off the combination of the elements of the reward vector. In contrast, in an FTSG, each agent only aims to maximize its own reward and the agents can be adversarial.

Stochastic games: Solving (both finite-horizon and infinite-horizon) N player stochastic games [Shapley, 1953] is a challenging task, especially under the Reinforcement Learning (RL) setting. Most work under the RL setting concentrates on two-player zero-sum stochastic games (TZSG) [Lagoudakis and Parr, 2002, Perolat et al., 2015]. Recently UCSG [Wei et al., 2017] extends techniques in MDP to TZSGs and gives a sample complexity of $\tilde{O}(\mathrm{poly}(1/\epsilon))$. Our work considers the finite-horizon turn-based games and presents a first provable framework, which enjoys comparable performance to the algorithms for MDPs.

Extensive games: Extensive games represent another type of sequential games in MARL. Fictitious Self-play (FSP) [Heinrich et al., 2015], an important method in extensive games, uses a naive exploration strategy similar to ϵ-greedy. Monte Carlo Counterfactual Regret Minimization with Outcome Sampling (MCCFROS) [Lanctot et al., 2009], which can be applied to solve extensive games (TZEG) under the RL setting, suffers from high variance. CFR-PSRL [Zhou et al., 2020] gives a provable solution for TZEGs under the Bayesian setting. Methods for extensive games may suffer from redundant calculation when used in stochastic games as they usually learn policies for histories rather than states.

[2]In this work, we follow [Dann et al., 2017] and consider time-dependent dynamics. We list the results in our setting.

Monte Carlo Tree Search (MCTS): MCTS [Coulom, 2006] is an efficient forward-search method which can be applied to turn-based games. MCTS combining with OFU exploration and estimated Q values reaches super-human performance in game Go [Silver et al., 2017a]. Since MCTS finds solutions by forward search, it can be myopic and cannot approach SPE solutions for games with large horizon.

3 PROBLEM FORMULATION

In this section, we formally define Finite-horizon Turn-based Stochastic Games (FTSG), the Nash Equilibrium (NE) and the performance measurements in FTSGs.

3.1 Finite-horizon Turn-based Stochastic games

We concentrate on games with a reset action. That is, the environment will reset to an initial state after a fixed number of time steps. We use *episode* to describe the steps between one initial state and its next reset state, and use *depth* to describe the steps from the initial state of the current episode.

Formally, a *Finite-horizon Turn-based Stochastic game* (FTSG) is a six-tuple $\mathcal{G} = \langle N, \mathcal{S}, \mathcal{A}, \mathcal{R}, P, H\rangle$:

- N is the number of players. We use $[N] = \{1, 2, ..., N\}$ to denote the player set.
- H is the largest depth of the game (i.e. the horizon).
- $\mathcal{S}$ is the state space with size S. For player $i \in [N]$, $\mathcal{S}_i \subset \mathcal{S}$ is its state space and $\mathcal{S}_i \cap \mathcal{S}_j = \emptyset$ if $i \neq j$.
- $\mathcal{A}$ is the action space with size A and $\mathcal{A}_i$ is the action space of player $i \in [N]$. Thus, $\mathcal{A} = \cup_{i\in[N]}\mathcal{A}_i$.
- Reward function $\mathcal{R}$ maps each state-action-depth tuple $(s, a, h) \in \mathcal{S} \times \mathcal{A} \times [H]$ to a probability distribution over $[0, 1]^N$. We use $R(s, a, h)$ to represent one vector sampled from the distribution and $R_i(s, a, h)$ represents the sampled reward for player i. We denote $r(s, a, h)$ (a vector) as the expectation of $R(s, a, h)$.
- $P(\cdot|s, a, h)$ is the transition probability over $\mathcal{S}$ from state s, action a and depth h. Here we consider a general time-dependent dynamics and thus P depends on h. We further use $P(s, a, h)$ to denote the transition vector for state s, action a and depth h.

We have no more assumptions on the game. The player order is not pre-defined but decided by the dynamics. Therefore, the FTSG class can include a large number of problems. For instance, if $\mathcal{S}_i = \mathcal{S}$, FTSG reduces to MDP. For convenience, we assume that the game begins from a specific state $s_1 \in \mathcal{S}$. The extension to random initial states is straightforward.

The policy of player i ($i \in [N]$), denoted as π_i, maps each state $s \in \mathcal{S}_i$ and its current depth h to an action $a \in \mathcal{A}_i$. We use Π_i to denote the set of all possible policies for π_i. The policies we define here are deterministic.[3] In one episode of the game, players follow $\pi = (\pi_1, ...\pi_N)$. Further, we use π_{-i} to denote the policy tuple that removes π_i from π. For notation clarity, we denote $\pi(s, h) = \pi_i(s, h)$ if $s \in \mathcal{S}_i$. For the kth episode, we denote the policy tuple we use as $\pi^k = (\pi_1^k, \pi_2^k, ..., \pi_N^k)$.

Following the common convention as in MDPs, we use V and Q values to represent the expected rewards for states and state-action pairs. For player $i \in [N]$, depth $h \in [H]$, state $s \in \mathcal{S}$ and action a for s, we define:

$$V_{i,h}^{\pi}(s) := \mathbb{E}\left[\sum_{h'=h}^{H} r_i(s_{h'}, \pi(s_{h'}, h'), h')|s_h = s\right],$$

$$Q_{i,h}^{\pi}(s, a) := \mathbb{E}\left[\sum_{h'=h}^{H} r_i(s_{h'}, \pi(s_{h'}, h'), h')|s_h = s, a_h = a\right],$$

where $s_{h'}$ is the state at depth h'. Note that even if $s \notin S_i$, we also define V and Q for player i. Further, we use $V_{i,h}^{\pi}$ without indicating the state to represent the vector for all states of horizon h.

The Bellman equation for the FTSG is

$$\begin{aligned} V_{i,h}^{\pi}(s) &= Q_{i,h}^{\pi}(s, \pi(s, h)) \\ &= r_i(s, \pi(s, h), h) + P(s, \pi(s, h), h)^{\top} V_{i,h+1}^{\pi}, \end{aligned}$$

for $h \in [H]$. Specifically, we define $V_{i,H+1}^{\pi}(s) = 0$ for all $i \in [N]$, $s \in \mathcal{S}$ and any π. We also define s_{H+1} as a terminal state for the convenience of notation.

3.2 Nash Equilibrium for Stochastic games

With the above definition, we now introduce the general learning goal for FTSGs, Nash Equilibrium (NE), and subgame perfect equilibrium (SPE), a refinement of NEs.

Definition 1. *A policy tuple $\pi^* = (\pi_1^*, \pi_2^*, ..., \pi_N^*)$, where $\pi_i^* \in \Pi_i$, $i \in [N]$, is a **Nash Equilibrium** (NE) of FTSG $\mathcal{G}$ if for all i and any $\pi_i' \in \Pi_i$,*

$$V_{i,1}^{\pi^*}(s_1) \geq V_{i,1}^{(\pi_i', \pi^*_{-i})}(s_1).$$

Definition 2. *A policy tuple $\pi = (\pi_1, \pi_2, ..., \pi_N)$, where $\pi_i \in \Pi_i$, $i \in [N]$, is an **ϵ-Nash Equilibrium** (ϵ-NE) of FTSG $\mathcal{G}$ if for any $\pi_i' \in \Pi_i$,*

$$V_{i,1}^{\pi}(s_1) \geq V_{i,1}^{(\pi_i', \pi_{-i})}(s_1) - \epsilon.$$

[3]They are usually called pure strategies in game theory.

A refinement of NE is the Subgame Perfect Equilibrium (SPE) [Nisan et al., 2007] where π is also a NE for any $s \in \mathcal{S}$ and depth $h \in [H]$. The SPE is a more suitable solution for FTSGs, since each FTSG has at least one deterministic policy tuple that is SPE. This can be proved with backward induction as in perfect-information extensive games [Osborne and Rubinstein, 1994]. However, this backward induction can only be conducted with full knowledge of the environment functions. In the RL setting, players must interact with the environment to explore the unknown transitions and rewards.

3.3 Performance Measurement

Our goal for learning is to find approximate NEs for FTSGs. Therefore, we define our performance measurement based on the concept of NE. In the analysis part we will show that the solution of our method has close relationship with SPEs.

We define the number of episodes in which the algorithm does not choose ϵ-NE as a performance measure:

$$L^{\epsilon} = \sum_{k \in \mathbb{N}} \mathbb{I}\left\{\pi^k \text{ is not an } \epsilon\text{−NE}\right\}.$$

By upper bounding L^{ϵ}, we measure the sample complexities of algorithms.

We further use exploitability [Lanctot et al., 2009], a closely related concept with approximate NE, to measure the peformance of algorithms over time. We define the cumulative exploitability for player i up to time T as

$$Expl_i(T) = \sum_{t=1}^{T} \max_{\pi_i} V_{i,1}^{(\pi_i, \pi_{-i}^t)}(s_1) - \sum_{t=1}^{T} V_{i,1}^{\pi^t}(s_1).$$

Then we define the total exploitability as

$$Expl(T) = \sum_{i \in [N]} Expl_i(T).$$

Our definition of $Expl$ corresponds to the regret in MDP. Similarly, we use high-probability exploitability bound as the performance measurement up to time T.

4 Exploration in FTSG

In this section, we analyze the challenges on exploration in FTSG in order to motivate our methods.

4.1 Exploration in MDP

We start by reviewing the key insight of exploration in single-player MDPs, for which Azar et al. [2017] have proposed efficient exploration algorithms following the *Optimism in the Face of Uncertainty* (OFU) principle. The essential idea of OFU is to choose the policy with maximum expected rewards under the current estimation for the true model. The upper bounds of Q values for state-action pairs can be calculated by backward induction. As illustrated in Fig. 1(a), we consider an MDP with 3 states and deterministic transitions. Rewards are only given at terminal nodes. We use colorful bars to indicate the confidence set for the optimal V values of nodes. Then the OFU principle chooses the actions with highest upper bounds. Therefore, the chosen policy for this iteration follows the trajectory of the yellow bar.

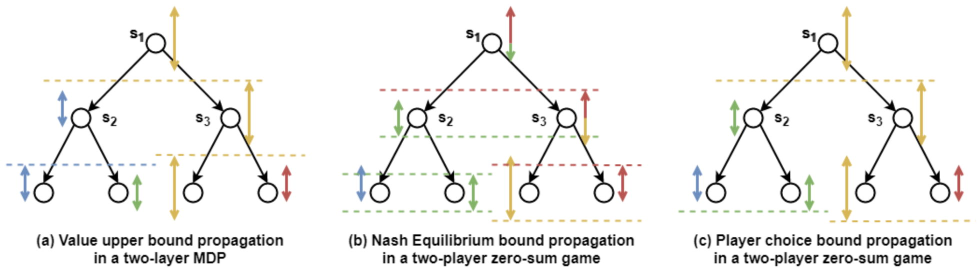
4.2 NE UNCERTAINTY ESTIMATION

It is nontrivial to extend the above OFU insight to stochastic games since the environment for each player is no longer static. We cannot identify the optimal policy for a player without taking other players into consideration. A straightforward idea is to estimate the uncertainty of the NE values and choose the optimal policy tuple accordingly. However, the chosen policy under this NE uncertainty estimation fails to explore efficiently since players cannot reach optimality simultaneously.

We use Fig. 1 (b) for an illustration. Consider a two-player zero-sum FTSG with 3 states and horizon 2. Player 1 acts at s_1 and player 2 acts at s_2 and s_3. Then a sampled reward for player 1 is returned. The color bars indicate the confidence bounds of NE values for player 1. The uncertainty of the four terminal nodes can be calculated and we get the NE bounds for the other nodes via back-propagation. For example for state s_3, player 2 here always chooses the minimum value of its two actions, and thus the NE value for s_3 is bounded by the minimum of upper and lower bounds of its children, i.e. the upper bound of red bar and the lower bound of yellow bar. However, such uncertainty estimation of NEs fails to guide exploration. Following the OFU principle, player 1 should choose the action on the right for the largest NE value and then player 2 should choose the left for s_3. However, we can see that the yellow bar does not directly contribute to the NE value estimation of s_1. That is, the trajectory in this episode is not the one we aim to explore. This happens because the two players cannot reach their optima at the same time in the face of uncertainty. This mismatch cannot guarantee efficient convergence to NEs.

4.3 Reaching optimality simultaneously

With the above observations, we realize that players need to reach optima at the same time such that they can gain the desired information for exploration. Notice that in FTSGs, the player at some depth can exactly infer the best choices for the nodes below it. Thus this player can choose its optimal action conditioning on the choices of



s1
s2
s3
(a) Value upper bound propagation in a two-layer MDP
(b) Nash Equilibrium bound propagation in a two-player zero-sum game
(c) Player choice bound propagation in a two-player zero-sum game

Figure 1: A simple example with 3 states (i.e., the decision points), 2 actions and horizon 2 to illustrate the exploration.

below states. In this way, we are able to design proper uncertainty estimations for players.

We revisit the game in Sec 4.2. As shown in Fig. 1 (c), player 2 should choose the green bar for s_2 and the yellow bar for s_3 since they have smaller lower bounds. The two bars are back-propagated to s_1 and player 1 chooses the yellow. By operating like this, each player reaches its optimum conditioning on the choices of the other. Hence they gradually converge to a NE. Furthermore, this idea makes no assumption on NEs and can be easily extended to N-player FTSGs. Based on this insight, we present our algorithm as well as the analysis in the next section.

5 OUR METHOD

We now present our framework, *Upper Bounding the Values for Players* (UBVP), as well as an analysis.

5.1 UBVP procedure

UBVP is motivated by the *Optimism in the Face of Uncertainty* (OFU) principle to conduct efficient exploration to find an approximate NE for FTSGs. The key part for exploration is to estimate the uncertainty of the values and design proper policies to interact with the environment. As analyzed in Sec. 4, we aim to find policies for players such that they reach optima conditioning on the policies of the other players. In order to do so, UBVP upper bounds the values of states based on the actions of states from deeper depths. UBVP is applicable to different implementations for the value estimations in order to satisfy different learning goals.

For convenience, we use $s_h^{k'}$, $a_h^{k'}$ and $R_h^{k'}$ to respectively denote the reached state, corresponding action and immediate reward at depth h of episode k'. Before episode k, the set of observed data is defined as $\mathcal{H}^k := \{(s_h^{k'}, a_h^{k'}, R_h^{k'}, s_{h+1}^{k'}) : h \in [H], k' \in [k-1]\}$. With $\mathcal{H}^k$, we calculate the count of visiting the state-action-depth tuple (s,a,h), denoted by $n^k(s,a,h)$, and the count of transiting to state s' immediately from (s,a,h), denoted by $n^k(s,a,s',h)$. We have $n^k(s,a,h) = \sum_{s'} n^k(s,a,s',h)$. For (s,a,h) at episode k with $n^k(s,a,h) > 0$, we have

$$\bar{r}_i^k(s,a,h) = \sum_{k' \in [k-1]} R_h^{k'} \mathbb{I}(s_h^{k'} = s, a_h^{k'} = a)/n^k(s,a,h), \tag{1}$$

$$\bar{P}^k(s'|s,a,h) = n^k(s,a,s',h)/n^k(s,a,h), \tag{2}$$

where s' is any possible next state.

Algorithm 1 Upper Bounding the Values for Players
```
 1: Input: $N, \mathcal{S}, \mathcal{A}, H, \mathcal{H}^1 = \emptyset, \delta$
 2: for episode $k = 1, 2, ...$ do
 3:   for $h = H, H-1, ..., 1$, $s \in \mathcal{S}$ do
 4:     for all possible actions $a \in \mathcal{A}$ and $i \in [N]$ do
 5:       Compute $\bar{r}_i^k(s,a,h)$ and $\bar{P}^k(s'|s,a,h)$ with
          Eq. (1) and (2) for possible $s'$
 6:       Compute $\tilde{Q}_{i,h}^k(s,a) = ComputingQ$
 7:     end for
 8:     $j = Player(s)$
 9:     $\pi^k(s,h) = \pi_j^k(s,h) = \arg\max_a Q_{j,h}^k(s,a)$
10:     $V_{i,h}^k(s) = Q_{i,h}^k(s, \pi^k(s,h))$ for all $i \in [N]$
11:   end for
12:   for step $h = 1, ..., H$ do
13:     Choose action $a_h^k = \pi^k(s_h^k, h)$
14:     Get to state $s_{h+1}^k$ and get reward vector $R_h^k$
15:   end for
16:   Update $\mathcal{H}^{k+1} = \mathcal{H}^k \cup \{(s_h^k, a_h^k, R_h^k, s_{h+1}^k)\}_{h=1}^H$
17: end for
```

Algorithm 2 ComputingQ
```
1: Input: $V_{i,h+1}^k$, $\bar{P}^k(s,a,h)$, $\bar{r}_i^k(s,a,h)$,
   $\{n^k(s,a,s',h)\}_{s'}, H, S, \delta$
2: $b_h^k(s,a) = \phi(\{n^k(s,a,s',h)\}_{s'}, \bar{P}(s,a,h), V_{i,h+1}^k, \delta)$
3: $Q_{i,h}^k(s,a) = \min\{Q_{i,h}^{k-1}(s,a), H, \bar{r}_i(s,a,h) +$
   $\bar{P}^k(s,a,h)^\top V_{i,h+1}^k + b_h^k(s,a)\}$
4: Output: $Q_{i,h}^k(s,a)$
```

Then we calculate the upper bounds of Q and V values from depth H to 1. Specifically, for each player i, with the calculated V-value upper bounds at depth $h+1$, denoted by $V^k_{i,h+1}$, we upper bound $Q^k_{i,h}(s,a)$ by adding $\bar{r}_i(s,a,h)+\bar{P}^k(s,a,h)^\top V^k_{i,h+1}$ with an extra bonus term $b^k_h(s,a) = \phi(\{n^k(s,a,s',h)\}_{s'}, \bar{P}(s,a,h), V^k_{i,h+1}, \delta)$. Here b^k_h is the bonus to bound the uncertainty of the estimation for the current Q value, and ϕ is a bonus function which can be defined in different ways to satisfy different performance measurements. We require it to satisfy the following property:

Property 1. *The bonus function ϕ for UBVP should satisfy that with a probability at least $1-\delta$, for all $i \in [N]$, $k > 0$, $h \in [H]$ and $(s,a) \in \mathcal{S} \times \mathcal{A}$:*

$$|(r^k_i - \bar{r}^k_i)(s,a,h) + (P^k - \bar{P}^k)(s,a,h)^\top V^{*,k}_{i,h+1}| \le b^k_h(s,a),$$

where $V^{*,k}_{i,h+1}(s) = \max_{\pi_i \in \Pi_i} V^{(\pi_i,\pi^k_{-i})}_{i,h+1}(s)$.

This property of ϕ can ensure that our calculated $Q^k_{i,h}(s,a)$ is a proper upper bound for $\max_{\pi_i} Q^{(\pi_i,\pi^k_{-i})}_{i,h}$, as we will prove in Lemma 2. This is the key that UBVP can guide proper exploration and converge to the NE.

The pseudo code of the UBVP algorithm is given in Alg. 1. Specifically, in line 8, we use $Player(s)$ to show the player to take action at state s. From line 3 to line 15, we calculate the upper bounds of V values for all N players through backward induction. At the same time, we work out the policy tuple π^k by letting each player choose the optimal action. This tuple is then used to play the game of this episode and collect data. The upper bounds calculation for Q values are given in Alg. 2.

5.2 Analysis for UBVP

Up to now, we have presented the framework of UBVP, except for the exact form of ϕ. Here based on the property of ϕ, we give a general analysis outline.

Firstly, we define the **best response distance** to describe how close a policy tuple is to the NE.

Definition 3. *For a policy tuple $\pi = (\pi_1, \pi_2, ..., \pi_N)$ where $\pi_i \in \Pi_i$, $i \in [N]$, the* ***best response distance (Bsd)*** *of player i for π is defined as*

$$Bsd_i(\pi) := \max_{\pi'_i \in \Pi_i} V^{(\pi'_i,\pi_{-i})}_{i,1}(s_1) - V^{(\pi)}_{i,1}(s_1).$$

Intuitively, $Bsd_i(\pi)$ measures the difference between the largest expected value that player i can get against π_{-i} and the actual value with π_i. Recall the definition of NE, it is natural to connect this value with ϵ-NE.

Lemma 1. *For policy tuple $\pi = (\pi_1, \pi_2, ..., \pi_N)$ where $\pi_i \in \Pi_i$, $i \in [N]$, if $Bsd_i(\pi) \le \epsilon$ for all player i, then π is an ϵ-NE.*

Although the proof is straightforward, we give it in Appendix A for completeness.

With Bsd_i, we now bound L^ϵ and $Expl(T)$ with

$$L^\epsilon \le \sum_{k\in\mathbb{N}^+} \mathbb{I}\left[\exists i \in [N], Bsd_i(\pi^k) > \epsilon\right],$$
$$Expl(T) = \sum_{i\in[N]} \sum_{k\in[T]} Bsd_i(\pi^k),$$

where $\mathbb{N}^+ := \{1, 2, ...\}$ is the set of positive integers. Therefore, we can turn to analyze Bsd_i to measure the performance of UBVP.

It is easy to see that in UBVP, players are symmetric, and thus the result for one can be adapted to the others. Now we consider player $i \in [N]$ and give the key lemma that connects the upper bound V and Bsd_i.

Lemma 2. *With a probability at least $1-\delta$, for UBVP with bonus function satisfying Property 1, for any $i \in [N]$, $k \in \mathbb{N}$, $h \in [H]$ and $s \in \mathcal{S}$,*

$$\max_{\pi_i \in \Pi_i} V^{(\pi_i,\pi^k_{-i})}_{i,h}(s) \le V^k_{i,h}(s).$$

This lemma can be proved by using Property 1 and induction on h. More specifically, we can first prove this result for $h = H$ and then use induction to prove all h. We complete this proof in Appendix B.

This is the key that UBVP can conduct exploration and converge to a NE. Intuitively, this is because we construct $V^k_{i,h}$ such that it is the upper bound of the optimal V for player i conditioning on other players' policies. Besides, $V^{(\pi_i,\pi^k_{-i})}_{i,h}(s)$ has the same action with π^k on states not in $\mathcal{S}_i$ and thus we ensure the upper bound property. Furthermore, the inequalities hold for all players at the same time, and this ensures them to converge to the best response of other players. This is exactly why we can converge to NEs.

Now we define $\Delta^k_i := V^k_{i,1}(s_1) - V^{\pi^k}_{i,1}(s_1)$. Using Lemma 2, with high probability, we have

$$Bsd_i(\pi^k) \le \Delta^k_i.$$

Then we turn to upper bound the sample complexity of Δ^k_i. Notice that the two terms in Δ^k_i follow the same policy tuple π^k but are calculated under different FTSGs. Inspired by techniques in MDP, we can decompose Δ^k_i and upper bound the separated terms.

For clarity, we use x to denote a state-action pair (s,a). For episode k and depth h, we use $w^k_h(x)$ to denote the probability of reaching state-action pair x at depth h following policy π^k. Now we can decompose Δ_i^k as

$$\begin{aligned}&V_{i,1}^k(s_1) - V_{i,1}^{\pi^k}(s_1)\\ =&\bar{r}_i^k(s_1,\pi(s_1,1),1) - r_i(s_1,\pi(s_1,1),1) + b_1^k(s_1,\pi(s_1,1))\\ &+ \bar{P}^k(s_1,\pi(s_1,1),1)^\top V_{i,2}^k - P(s_1,\pi(s_1,1),1)^\top V_{i,2}^{\pi^k}\\ =&\sum_{h=1}^{H}\sum_{x\in\mathcal{S}\times\mathcal{A}} w_h^k(x)\big((\bar{r}_i^k - r_i)(x,h)\\ &+ \left(\bar{P}^k - P\right)(x,h)^\top V_{i,h+1}^k + b_h^k(x)\big). \qquad (3)\end{aligned}$$

Therefore, we can design a proper bonus function ϕ such that Eq. (3) can be upper bounded. This bound for Δ_i^k is also the bound for $Bsd_i(\pi^k)$. And finally, we can get the upper bound for either $Expl(T)$ or L^ϵ.

The specific theoretical analysis can vary for different forms of ϕ and different performance measurements. Fortunately, there exit various value-based algorithms on MDPs in UCB-type. Many of them can be be adapted to UBVP. Sec. 6 shall present two examples.

5.3 Relationship with SPE

We have shown above that UBVP can approach an NE. In fact, UBVP is also approaching a SPE, a refined concept of NE. This can be shown by considering Lemma 2. Notice that this lemma holds for all s and h with a high probability. Recall that we use backward induction to calculate policies. Therefore for each time we reach state s at depth h, UBVP also solves the NE of this subgame. At the same time, since efficient exploration is required, UBVP allocates more resources to subgames with higher utilities. Hence the solution of UBVP is indeed an approximated SPE, while this solution might have relatively high approximate errors on low utility subgames.

6 TWO ALGORITHMS ON UBVP

As analyzed above, we can design different ϕ for different performance measurements. Below we give two concrete examples for UBVP. The first is a Uniform-PAC algorithm with L^ϵ bounded by $\tilde{O}(1/\epsilon^2)$ and the second has a cumulative exploitability of order $\tilde{O}(\sqrt{T})$ with high probability. The former is suitable for cases where we aim to identify an approximate NE and the latter can be used when we wish to get small exploitability over time.

6.1 A Uniform-PAC algorithm

Uniform-PAC [Dann et al., 2017] is a framework describing the sample complexity of an algorithm to reach an approximate solution with high probability. Formally, an algorithm is Uniform-PAC if with a probability at least $1-\delta$, for all $\epsilon > 0$, L^ϵ is upper bounded by some function $f^{UPAC}(N,S,A,H,\epsilon,\delta)$. Here we follow the design of UBEV in [Dann et al., 2017] and choose the bonus term ϕ^{UPAC} as:

$$\phi^{UPAC} = (H+1)\sqrt{\frac{2\ln\ln(\max\{e, n^k(s,a,h)\}) + \tau)}{n^k(s,a,h)}},$$

where $\tau = \ln((24N+30)SA/\delta$. Following the proof of Corollary E.1 in [Dann et al., 2017], we verify ϕ^{UPAC} satisfies Property 1 . We give a proof in the Appendix C.

We give the sample complexity for UBVP with ϕ^{UPAC}.

Theorem 1. *(Uniform-PAC bound) If UBVP chooses the bonus function as ϕ^{UPAC}, then for $\delta \in (0,1)$ and $\epsilon > 0$, with a probability at least $1-\delta$, L^ϵ is upper bounded by*

$$O\left(\frac{H^4SA}{\epsilon^2}\min\{S, N+\epsilon(N+S^2A)\}\mathcal{L}\right),$$

where $\mathcal{L} = \text{polylog}(N,H,S,A,1/\delta,1/\epsilon)$.

The proof for this theorem is given in Appendix C, which is inspired by the analysis of Dann et al. [2017].

The complexity of UBVP on FTSGs is comparable with the related work on MDPs. Specifically, the previous Uniform-PAC algorithm for MDPs, UBEV, achieves the sample complexity upper bound of $O(\frac{H^4SA}{\epsilon^2}\min\{S, 1+\epsilon S^2A\}\text{polylog}\left(H,S,A,\frac{1}{\delta},\frac{1}{\epsilon}\right))$. Therefore, for relatively large ϵ, UBVP solves FTSG with a sample complexity the same as that of UBEV, except for an extra N in logarithmic terms. Considering that UBVP works out a policy tuple with N policies as the solution for FTSG, the extra cost on N is indeed cheap. For a fixed $\epsilon > 0$, the lower bound of the sample complexity for MDPs is $\tilde{\Omega}(\frac{H^3SA}{\epsilon^2}\ln(\frac{SA}{\delta}))$ [Dann et al., 2017] for sufficiently small ϵ. This lower bound is also suitable for FTSGs.

6.2 An algorithm with High Probability Exploitability Bound

For this case, we choose two kinds of bonus functions:

$$\begin{aligned}\phi_1^{HPR} &= 8H\tau'\sqrt{1/n^k(s,a,h)},\\ \phi_2^{HPR} &= \sqrt{\frac{8\tau' Var_{s'\sim\bar{P}(s,a,h)}V_{i,h+1}^k(s')}{n^k(s,a,h)}} + \frac{14H\tau'}{3n^k(s,a,h)}\\ &+ HL\sqrt{\frac{1}{n^k(s,a,h)}} + \sqrt{\frac{8\sum_{s'}\bar{P}(s,a,s',h)C(s')}{n^k(s,a,h)}},\end{aligned}$$

where $C(s') = \min\{10^4H^3S^2A\tau'^2/n^k(s,a,s',h), H^2\}$ and $\tau' = \ln(5HSATN/\delta)$. The two ϕ functions are designed by extending UCB-VI [Azar et al., 2017]. We add $H\tau'\sqrt{1/n^k(s,a,h)}$ to bonuses of UCB-VI to design ϕ. The extra term is designed to upper bound reward functions. The two kinds of ϕ satisfy Property 1 using the analysis of Lemma 18 and Sec.5.2 in [Azar et al., 2017]. Then we have:

Theorem 2. *(High Probability Exploitability Bound) If UBVP uses ϕ_1^{HPR} as its bonus function, then with a probability at least $1-\delta$, the cumulative exploitability $Expl(T)$ of UBVP has an order of:*

$$O\left(NH\tau'\sqrt{HSAT} + H^3S^2A\tau'^2\right).$$

If UBVP uses ϕ_2^{HPR} as its bonus function, then with a probability at least $1-\delta$, the cumulative exploitability $Expl(T)$ of UBVP has an order of:

$$O\left(N\tau'H\sqrt{SAT} + NH^3S^2A\tau'^2 + NH\sqrt{T\tau'}\right).$$

The proof is straightforward by combining Lemma 2 with the analysis in [Azar et al., 2017]. Notice that the setting in [Azar et al., 2017] is time-independent dynamics. Thus we only need to replace S with SH when the pigeon-hole principle is used.

6.3 Other extensions

We have presented two concrete examples of UBVP. In fact, many other algorithms on MDP can be adapted to UBVP. For example, recent work provides tighter bounds by considering the lower bounds for Q values [Dann et al., 2019, Zanette and Brunskill, 2019]. Then we can build similar lower bounds for $Q_{i,h}^k$ and $V_{i,h}^k$ in UBVP:

$$\begin{aligned}\underset{\sim}{Q}_{i,h}^k(s,a) &= \min\{Q_{i,h}^{k-1}(s,a), H,\\ &\qquad \bar{r}_i(s,a,h) + \bar{P}^k(s,a,h)^\top \underset{\sim}{V}_{i,h+1}^k - b_h^k(s,a)\},\\ \underset{\sim}{V}_{i,h+1}^k(s) &= \underset{\sim}{Q}_{i,h}^k(s,\pi^k(s,h)).\end{aligned}$$

Then we can still choose ϕ following existing work [Dann et al., 2019, Zanette and Brunskill, 2019] to give the corresponding exploitability bounds or PAC sample complexity. Notice that these designs of ϕ highly rely on the estimated Q values and thus the exploitability bound or sample complexity should be N times the regret bound or sample complexity in MDP.

7 EXPERIMENTS

We now provide empirical evaluation. We compare the performances of algorithms under three game settings.

We choose the average exploitability $Expl(t)/t$ as our performance measure. Baselines are state-of-the-art methods in MARL. Specifically, FSPFQI [Heinrich et al., 2015], MCCFROS [Lanctot et al., 2009] and CFR-PSRL [Zhou et al., 2020] are RL methods for extensive games. MCTS [Coulom, 2006] is an efficient forward search method and NashQ [Hu and Wellman, 2003] is an important method for stochastic games. For UBVP, we choose ϕ_1^{HPR} as our bonus function. We test each algorithm for 10 times in each setting. More details of the implementation and games are deferred to Appendix E.

7.1 Two-player zero-sum FTSG

In a two-player zero-sum FTSG, players take actions alternately and there are only two states at each depth. Each state-action pair has a non-zero probability to transit to the two states at next depth. We design this game such that rewards are only generated at the last depth. We generate 10 games with random transitions and rewards, and test our methods on them.

We choose the horizon to be 4. The average exploitabilities are shown in Fig. 2(a). We can see that only MCTS and UBVP efficiently decrease the exploitability.

7.2 Cooperative two-player FTSG

The second experiment is on a more difficult game where sufficient exploration is needed. We design a tree game with deterministic transitions. Each state has two actions and two players have the same rewards. For the initial state s_1, if its player chooses action a_1, the expected rewards for both players will always be 0.5. If the player at s_1 chooses a_2, the expected rewards for both players are 0.4 except only one trajectory, which gets an expected reward of 0.6. The reward function is a Bernoulli function. We set $H = 8$. Here the SPE value is the optimal solution and we specifically use $Expl(T) = 0.6T - \sum_{t=0}^{T} V_{0,1}^t$ as the SPE exploitability.

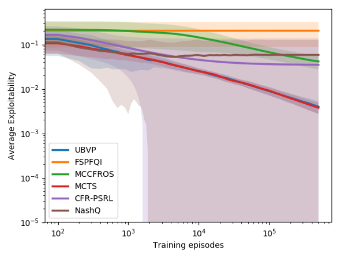
Fig. 2(b) shows the results. We can see that FSPFQI and MCCFROS are decreasing quite slowly. MCTS, NashQ and CFR-PSRL converge to a sub-optimal NE and only UBVP can find the optimal SPE solution.

7.3 Multi-player FTSG

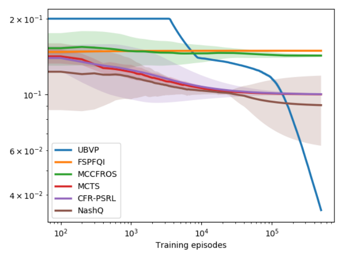
We also test the methods on a multi-player FTSG. We test methods on different kinds of multi-player games. The games have the same structures as the two-player zero-sum FTSG in Sec. 7.1, except that the expected reward for each player at the terminal nodes is drawn from a uniform distribution.

Methods are tested for ten times on randomly generated games. We choose three kinds of games with $(N,H) \in \{(3,6),(4,6),(3,8)\}$. We show the averaged $expl$ and computation time for algorithms when $T = 100000$ in Table 1. For game with $(N,H) = (3,6)$, the average $expl$ is shown in Fig. 2(c). UBVP has comparable performance with state-of-the-art methods MCTS. Notice CFR-based methods are very slow since they need to consider each history separately. Other methods lack efficient exploration strategies and have poor performance.

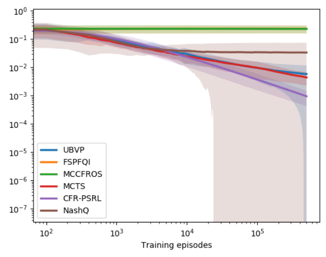
These empirical results demonstrate that UBVP has competitive performance with state-of-the-art methods,



(a) Adversarial FTSG



(b) Cooperative FTSG



(c) Three-player FTSG

Figure 2: Average $Expl(t)/t$ over t of UBVP and various baseline methods on three FTSG games.

Table 1: Averaged exploitablity and time when $T = 100000$. For each game (N, H), the above line is the averaged exploitablity and the below line is the calculating time.

GAMES	UBVP	FSPFQI	MCCFROS	MCTS	CFR-PSRL	NashQ
$(3,6)$	0.017 ± 0.022	0.19 ± 0.06	0.19 ± 0.06	0.010 ± 0.003	$\mathbf{0.004 \pm 0.002}$	0.031 ± 0.027
	73 ± 4	272 ± 18	105 ± 6	33 ± 2	321 ± 16	$\mathbf{31 \pm 2}$
$(3,8)$	0.017 ± 0.009	0.16 ± 0.07	0.17 ± 0.07	0.013 ± 0.004	$\mathbf{0.008 \pm 0.006}$	0.063 ± 0.061
	104 ± 2	378 ± 13	925 ± 19	43 ± 4	4571 ± 73	$\mathbf{40 \pm 1}$
$(4,8)$	0.027 ± 0.024	0.20 ± 0.09	0.20 ± 0.09	0.015 ± 0.010	$\mathbf{0.005 \pm 0.003}$	0.084 ± 0.053
	99 ± 1	363 ± 2	1103 ± 32	40 ± 1	3623 ± 54	$\mathbf{38 \pm 1}$

which shows that it can indeed efficiently explore the environment. At the same time, UBVP can guarantee sufficient exploration and approach the SPE solution.

8 DISCUSSION

As known in game theory, it is possible for a game to have more than one NE. The solution of UBVP can ensure that players converge to the same NE. Moreover, we can ensure that our solution is an approximation of an SPE, as analyzed in Sec. 5.3. If multiple SPEs exist, UBVP would explore them all by tightening their upper bounds. It may be possible for UBVP to recommend a specific SPE with some extra designs for policy chosen. We think this should be an interesting future work.

Our algorithm ensures the convergence to approximate NE only if all players follow UBVP. The exploration of the unknown environment requires all players to cooperate. Otherwise it is possible that some potential policies are not identified. Therefore if the learning goal is to efficiently identify an NE, it is necessary that all players can explore together. In the decentralized cases, each player aims rewards without considering other players. UCSG [Wei et al., 2017] provides a decentralized solution for two-player zero-sum stochastic games. Intuitively, we think it might be possible that there exist some centralized methods can also perform well, e.g. low regret, in the decentralized case.

9 CONCLUSIONS

We present UBVP to extend the OFU principle in MDP to N-player FTSGs. UBVP guides efficient exploration to converge to approximate NEs, and it is the first method that gives a non-asymptotic analysis of NEs for FTSG in the Reinforcement Learning setting. We propose two algorithms based on UBVP, which have Uniform-PAC and high probability exploitability guarantees, respectively. Our analysis shows that these algorithms match the results on finite-horizon MDPs except for the term N.

Stochastic games that are not turn-based or infinite-horizon are more complicated cases and the exploration for such games is still an open challenge. Our work could provide some insights for solving this challenge. Essentially, UBVP mainly provides a way for players to choose actions that can be optimal against others. We believe that this is also one of the key issues for these more general cases and is worthy of a systematic investigation.

Acknowledgement

This work was supported by NSFC Projects (Nos. 61620106010, U19B2034, U1811461), Beijing NSF Project (No. L172037), Beijing Academy of Artificial Intelligence (BAAI), and a grant from Tsinghua Institute for Guo Qiang.

References

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017a.

Richard Rouse III. *Game design: Theory and practice*. Jones & Bartlett Learning, 2010.

Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V Vazirani. *Algorithmic game theory*. Cambridge university press, 2007.

Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *arXiv preprint arXiv:1911.10635*, 2019.

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

Christoph Dann and Emma Brunskill. Sample complexity of episodic fixed-horizon reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 2818–2826, 2015.

Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pages 157–163. Elsevier, 1994.

Junling Hu and Michael P Wellman. Nash q-learning for general-sum stochastic games. *Journal of machine learning research*, 4(Nov):1039–1069, 2003.

Chen-Yu Wei, Yi-Te Hong, and Chi-Jen Lu. Online reinforcement learning in stochastic games. In *Advances in Neural Information Processing Systems*, pages 4987–4997, 2017.

Edward Lockhart, Marc Lanctot, Julien Pérolat, Jean-Baptiste Lespiau, Dustin Morrill, Finbarr Timbers, and Karl Tuyls. Computing approximate equilibria in sequential adversarial games by exploitability descent. *arXiv preprint arXiv:1903.05614*, 2019.

Johannes Heinrich, Marc Lanctot, and David Silver. Fictitious self-play in extensive-form games. In *International Conference on Machine Learning*, pages 805–813, 2015.

Marc Lanctot, Kevin Waugh, Martin Zinkevich, and Michael Bowling. Monte carlo sampling for regret minimization in extensive games. In *Proceedings of the 22nd International Conference on Neural Information Processing Systems*, pages 1078–1086, 2009.

Yichi Zhou, Jialian Li, and Jun Zhu. Posterior sampling for multi-agent reinforcement learning: solving extensive games with imperfect information. In *International Conference on Learning Representations*, 2020.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017b.

Christoph Dann, Tor Lattimore, and Emma Brunskill. Unifying pac and regret: Uniform pac bounds for episodic reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 5713–5723, 2017.

Diederik M Roijers, Peter Vamplew, Shimon Whiteson, and Richard Dazeley. A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research*, 48:67–113, 2013.

Lloyd S Shapley. Stochastic games. *Proceedings of the National Academy of Sciences*, 39(10):1095–1100, 1953.

Michail G Lagoudakis and Ronald Parr. Value function approximation in zero-sum markov games. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, pages 283–292. Morgan Kaufmann Publishers Inc., 2002.

Julien Perolat, Bruno Scherrer, Bilal Piot, and Olivier Pietquin. Approximate dynamic programming for two-player zero-sum markov games. In *International Conference on Machine Learning*, pages 1321–1329, 2015.

Rémi Coulom. Efficient selectivity and backup operators in monte-carlo tree search. In *Proceedings of the 5th international conference on Computers and games*, pages 72–83. Springer-Verlag, 2006.

Martin J Osborne and Ariel Rubinstein. *A course in game theory*. MIT press, 1994.

Christoph Dann, Lihong Li, Wei Wei, and Emma Brunskill. Policy certificates: Towards accountable reinforcement learning. In *International Conference on Machine Learning*, pages 1507–1516, 2019.

Andrea Zanette and Emma Brunskill. Tighter problem-dependent regret bounds in reinforcement learning without domain knowledge using value function bounds. In *International Conference on Machine Learning*, pages 7304–7312, 2019.