On-line Markov Decision Processes

Eyal Even-Dar*, Sham. M. Kakade†, Yishay Mansour‡

Abstract

We consider an MDP setting in which the reward function is allowed to change during each time step of play (possibly in an adversarial manner), yet the dynamics remain fixed. Similar to the experts setting, we address the question of how well can an agent do when compared to the reward achieved under the best stationary policy over time. We provide *efficient* algorithms, which have regret bounds with *no dependence* on the size of state space. Instead, these bounds depend only on a certain horizon time of the process and logarithmically on the number of actions.

1 Introduction

[1]Finite state and actions Markov Decision Processes (MDPs) are popular and attractive way to formulate many stochastic optimizations problems ranging from robotics to finance (Puterman [18] ; Bertsekas and Tsisiklis [3]; Sutton and Barto [19]). Unfortunately, in many application the Markovian assumption made is only a relaxation of the real model. A popular framework which is not Markovian is the experts problem, in which during every round a learner chooses one of n decision making experts and incurs the loss of the chosen expert. The setting is typically an adversarial one, where Nature provides the examples to a learner. The standard objective here is a myopic, backwards looking one — in retrospect, we desire that our performance is not much worse than had we chosen any *single* expert on the sequence of examples provided by Nature. Expert algorithms have played an important rule in Computer science in the last decade solving problems varying from classification to on-line portfolios Littlestone and Warmuth [14], Blum and Kalai [2] and Helmbold *et al.* [9].

There is an inherent tension between the objectives in an expert setting and those in a reinforcement learning (RL) setting. In contrast to the myopic nature of the expert algorithms, a reinforcement learning setting typically makes the much stronger assumption of a fixed environment, and the forward looking objective is to maximize some measure of the future reward with respect to this fixed environment. Therefore, in RL the past actions have

*University of Pennsylvania, Department of Computer and Information Science, email: evendar@seas.upenn.edu

†Toyota Technological Institute, email: sham@tti-c.org

‡School of Computer Science, Tel-Aviv University, email: mansour@post.tau.ac.il

[1]A prelimenry version of this paper appeared in NIPS 2004 [5]

a major influence on the current reward, while in regret setting they have no influence. In this paper we relax the Markovian assumption of the MDPs by letting the reward function being time dependent and even chosen by an adversary as is done in the expert setting, but still keeping the underlying structure of an MDP.

The motivation of this work is to understand how to *efficiently* incorporate the benefits of existing experts algorithms into a more adversarial reinforcement learning setting, where certain aspects of the environment could change over time. A naive way to implement an experts algorithm is to simply associate an expert with each fixed policy. The running time of such algorithms is polynomial in the number of experts and the regret (the difference from the optimal reward) is logarithmic in the number of experts. For our setting the number of policies is huge, namely $\#\text{actions}^{\#\text{states}}$, which renders the naive experts approach computationally infeasible. Another inherent problem in applying the best expert algorithms is that the current reward of the policy *depends* on the past actions, which is never the case in the standard expert setting. Furthermore, straightforward applications of standard regret algorithms produce regret bounds which are logarithmic in the number of policies, so they have linear dependence on the number of states. We might hope for a more effective regret bound which has *no dependence* on the size of state space (which is typically large).

The setting we consider is one in which the dynamics of the environment are known to the learner and are Markovian, but the reward function can change over time. We assume that after each time step the learner has full information, i.e. complete knowledge of the previous reward functions (over the entire environment), but does not know the future reward functions.

As a motivating example one can consider taking a long road-trip over some period of time T. The dynamics, namely the roads, are fixed, but the road conditions may change frequently. By listening to the radio, one can get (effectively) instant updates of the road and traffic conditions. Here, the task is to minimize the cost during the period of time T. Note that at each time step we select one road segment, suffer a certain delay, and need to plan ahead with respect to our current position.

This example is similar to an adversarial shortest path problem considered in Kalai and Vempala [11]. In fact Kalai and Vempala [11], address the computational difficulty of handling a large number of experts under certain linear assumptions on the reward functions. However, their algorithm is not directly applicable to our setting, due to the fact that in our setting, decisions must be made with respect to the *current* state of the agent (and the reward could be changing frequently), while in their setting the decisions are only made with respect to a single state. For our motivating example, their setting are when you decide at the beginning of each day on the entire route from a to b, while in our setting in every crossing you choose your next step in order to minimize the time to reach b.

1.1 Related work

McMahan *et al.* . [15] also considered a similar setting — they also assume that the reward function is chosen by an adversary and that the dynamics are fixed. However, they assume that the cost functions come from a finite set (but are not observable) and the goal is to find

a min-max solution for the related stochastic game.

De Farias and Megiddo [6, 7] considered similar problem but yet different. Similarly to our setting, they have both a "state", which is effected by the past actions of the expert, and the assumption that after following an expert for a while to past is "forgotten". A few notable differences are that we can evaluate each expert even if we do not follow him since we can estimate its state while in their works this is impossible as the current reward is influenced in the past in an unknown manner. Their main goal is to gain when the nature is not adversarial which is non issue in our setting. More importantly, if one implement their algorithm on in our setting it will be exponential in both time and space while our algorithm is efficient and exploit the extra knowledge that it is given.

Nilim and El-Ghaoui [17] studied robust MDPs, which have known rewards distributions and their transition matrices come from a convex set. They studied two possible models therein, one is that the transition matrix is stationary, chosen once by nature at the beginning and second where the nature chooses at each timestep a matrix from the set. For these setting they show how to compute the optimal policy using the linear programming methods. This can be thought as game where the adversary chooses the transition matrix from a given set each round.

2 The Model

The on-line MDP similarly to standard MDP consists of state space S; Actions available to the agent at each state A; A transition matrix P which specify the probability of arriving s' from s after performing a for every s, s' and a; initial state distribution d_1 over S; and a sequence of reward functions $r_1, r_2, \ldots r_T$, where r_t is the (bounded) reward function at time step t mapping $S \times A$ into $[0, 1]$.

The goal is to maximize the sum of *undiscounted* rewards over a T step horizon. We assume the agent has complete knowledge of the transition model P, but at time t, the agent only knows the past reward functions $r_1,\ r_2,\ \ldots r_{t-1}$. Hence, an algorithm $\mathcal{A}$ is a mapping from S and the previous reward functions $r_1, \ldots r_{t-1}$ to a probability distribution over actions, so $\mathcal{A}(a|s, r_1, \ldots r_{t-1})$ is the probability of taking action a at time t.

We define the return of an algorithm $\mathcal{A}$ as:

$$V_{r_1,r_2,\ldots r_T}(\mathcal{A}) = \frac{1}{T}\mathrm{E}\left[\sum_{t=1}^{T} r_t(s_t, a_t)\Big| d_1, \mathcal{A}\right]$$

where $a_t \sim \mathcal{A}(a|s_t, r_1, \ldots r_{t-1})$ and s_t is the random variable which represents the state at time t, starting from initial state $s_1 \sim d_1$ and following actions $a_1, a_2, \ldots a_{t-1}$. Note that we keep track of the expectation and not of a specific trajectory (and our algorithm specifies a distribution over actions at *every* state and at *every* time step t). This assumption is necessary since a specific trajectory makes the adversary too powerful, and also note that when we compute the value of the optimal policy we *must* look on its expected value, as we don't have a specific trajectory in hand.

Ideally, we would like to find an algorithm $\mathcal{A}$ which achieves a large reward $V_{r_1,\ldots r_T}(\mathcal{A})$ *regardless* of how the adversary chooses the reward functions. In general, this of course is not possible, and, as in the standard experts setting, we desire that our algorithm competes favorably against the best fixed stationary policy $\pi(a|s)$ in hindsight.

3 Mixing Time

Before we provide our results a few definitions are in order. For every stationary policy $\pi(a|s)$, we define P^π to be the transition matrix induced by π, where the component $[P^\pi]_{s,s'}$ is the transition probability from s to s' under π, i.e. $[P^\pi]_{s,s'} = \sum_{a \in A} \pi(a|s) P^a_{s,s'}$. Also, define $d_{\pi,t}$ to be the state distribution at time t when following π, i.e.

$$d_{\pi,t} = d_1 (P^\pi)^t$$

where we are treating d_1 as a row vector here.

We assume throughout this section that the MDP is a unichain Model.

Assumption 3.1 *(Mixing) Since we assume that the MDP is uni-chain every policy π has a well defined stationary distribution, which we call d_π. More formally, for every initial state s, $d_{\pi,t}$ converges to d_π as t tends to infinity and $d_\pi P^\pi = d_\pi$. Furthermore, this implies there exists some τ such that for* all *policies π, and distributions d and d',*

$$\|dP^\pi - d'P^\pi\|_1 \leq e^{-1/\tau} \|d - d'\|_1$$

where $\|x\|_1$ denotes the l_1 norm of a vector x. We refer to τ as the mixing time *and assume that $\tau > 1$.*

The parameter τ provides a bound on the planning horizon timescale, since it implies that *every* policy achieves close to its average reward in $O(\tau)$ steps [2]. This parameter also governs how long it effectively takes to switch from one policy to another (after time $O(\tau)$ steps there is little information in the state distribution about the previous policy). This definition is very similar to definition of flexibility made by De Farias and Megiddo.

This assumption allows us to define the average reward of policy π in an MDP with reward function r as:

$$\eta_r(\pi) = \mathrm{E}_{s \sim d_\pi, a \sim \pi(a|s)}[r(s,a)]$$

and the value, $Q_{\pi,r}(s,a)$, is defined as

$$Q_{\pi,r}(s,a) \equiv \mathrm{E}\left[\sum_{t=1}^{\infty} \left(r(s_t,a_t) - \eta_r(\pi) \right) \Big| s_1 = s, a_1 = a, \pi \right]$$

[2] If this timescale is unreasonably large for some specific MDP, then one could artificially impose some horizon time and attempt to compete with those policies which mix in this horizon time, as done by Kearns and Singh [12].

where and s_t and a_t are the state and actions at time t, after starting from state $s_1 = s$ then deviating with an immediate action of $a_1 = a$ and following π onwards. We slightly abuse notation by writing $Q_{\pi,r}(s, \pi') = \mathrm{E}_{a\sim\pi'(a|s)}[Q_{\pi,r}(s, a)]$. These values satisfy the well known recurrence equation:

$$Q_{\pi,r}(s, a) = r(s, a) - \eta_r(\pi) + \mathrm{E}_{s'\sim P_{sa}}[Q_\pi(s', \pi)] \tag{1}$$

where $Q_\pi(s', \pi)$ is the next state value (without deviation).

If π^* is an optimal policy (with respect to η_r), then, as usual, we define $Q^*_r(s, a)$ to be the value of the optimal policy, i.e. $Q^*_r(s, a) = Q_{\pi^*,r}(s, a)$.

We now provide two useful lemmas. It is straightforward to see that the previous assumption implies a rate of convergence to the stationary distribution that is $O(\tau)$, for all policies. The following Lemma states this more precisely.

Lemma 3.2 *For all policies π,*

$$\|d_{\pi,t} - d_\pi\|_1 \leq 2e^{-t/\tau} .$$

Proof: Since π is stationary, we have $d_\pi P^\pi = d_\pi$, and so

$$\|d_{\pi,t} - d_\pi\|_1 = \|d_{\pi,t-1}P^\pi - d_\pi P^\pi\|_1 \leq \|d_{\pi,t-1} - d_\pi\|_1 e^{-1/\tau}$$

which implies $\|d_{\pi,t} - d_\pi\|_1 \leq \|d_1 - d_\pi\|_1 e^{-t/\tau}$. The claim now follows since, for all distributions d and d', $\|d - d'\|_1 \leq 2$. ∎

The following Lemma derives a bound on the Q values as a function of the mixing time.

Lemma 3.3 *For all reward functions r in $[0, 1]$ and policies π, $Q_{\pi,r}(s, a) \leq 3\tau$.*

Proof: First, let us bound $Q_{\pi,r}(s, \pi)$, where π is used on the first step. For all t, including $t = 1$, let $d_{\pi,s,t}$ be the state distribution at time t starting from state s and following π. Hence, we have

$$\begin{aligned}
Q_{\pi,r}(s, \pi) &= \sum_{t=1}^{\infty} \left(E_{s'\sim d_{\pi,s,t},a\sim\pi}[r(s', a)] - \eta_r(\pi)\right) \\
&\leq \sum_{t=1}^{\infty} \left(E_{s'\sim d_\pi,a\sim\pi}[r(s', a)] - \eta_r(\pi) + 2e^{-t/\tau}\right) \\
&= \sum_{t=1}^{\infty} 2e^{-t/\tau} \leq \int_0^{\infty} 2e^{-t/\tau} = 2\tau
\end{aligned}$$

Using the recurrence relation for the values, we know $Q_{\pi,r}(s, a)$ could be at most 1 more than the above. The result follows since $1 + 2\tau \leq 3\tau$ ∎

4 Best Expert Algorithms

We first provide our assumption on the performance expert algorithms and later for completeness provide the weighted majority algorithm (Littlestone and Warmuth [14], Cesa-Bianchi *et al.* [4]).

Assumption 4.1 *(Black Box Experts) An optimized best expert algorithm is an algorithm which guarantees that for any sequence of loss functions $c_1, c_2, \dots c_T$ over actions A, the algorithm selects a distribution q_t over A (using only the previous loss functions $c_1, c_2, \dots c_{t-1}$) such that*

$$\sum_{t=1}^{T} \mathrm{E}_{a\sim q_t}[c_t(a)] \leq \sum_{t=1}^{T} c_t(a) + \sqrt{TM \log |A|},$$

where $\|c_t(a)\| \leq M$. Furthermore, we also assume that decision distributions do not change quickly:

$$\|q_t - q_{t+1}\|_1 \leq \sqrt{\frac{\log |A|}{t}}$$

Next we describe the weighted majority algorithm Littlestone and Warmuth [14] Ceas-Bianchi *et al.* [4], which satisfies this assumption, although many other popular algorithms as the exponential gradient (Kivinen andWarmuth [13])do satisfy the assumption as well. The WM algorithm variant that we present here requires knowing the horizon time T in advance, however there exists other variants such as doubling Cesa-Bianchi *et al.* [4] or changing the learning rate Auer *et al.* [1] that achieve the desired regret bounds without knowing T in advance.

Weighted Majority (T)
Choose an initial distribution P_1;
for $t = 1$ *to* T **do**
  Update $P_{t+1}(a) = P_t(a)\beta^{c_t(a)}/Z_t$, where $Z_t = \sum_a P_t(a)\beta^{c_t(a)}$;
end

Proposition 4.2 *For the weighted majority algorithm with appropriate β, we have*

$$E[\textit{Cost Wighted Majority at time } T] \leq \textit{Best expert at time } T + O(\sqrt{M \log |A|}T).$$

A different flavor of an best expert algorithm, which does not satisfy Assumption 4.1 but can be much more efficient, was introduced in Kalai and Vempala [11] Hanan [8]. The Follow the perturbed leader (FPL) algorithm works for problems that have the following properties (1) There exists an oracle (efficient method) which computes the optimum of the static problem, (2) The cost function is additive. The setting is defined as follows, a decision set such that for any two decisions we have $\|d - d'\|_1 \leq D$ (in the former setting each decision

was associated with an expert), state set S, cost function $c(d, s) \leq R$; the cost of each state is revealed at the end of each timestep.

FPL
for $t = 1$ *to* ∞ **do**
Choose p_t uniform at random in $[0, |S|\sqrt{t}]^{|S|}$;
Use an oracle M to find the optimal decision for cost function $\sum_{i=1}^{t} c_t + p_t$;
Use the optimal decision at time $t + 1$;
end

The following Proposition bounds the regret of FPL algorithm.

Proposition 4.3 *The FPL algorithm satisfies*

$$E[\textit{Cost of FPL at time } T] \leq c(d, s_1 + ... + s_T) + 2\sqrt{DR|S|T},$$

for every fixed decision d.

The major advantage of the FPL type of algorithms is that they can handle an exponential number of experts as long as the static problem can solved efficiently as is evident in the shortest path problem. In the shortest path problem an agent is given a graph with two special nodes s, t in each time step the agent chooses a path between s and t and weight edges are revealed. The agent wants to minimize its average path distance and the regret is measured with respect to optimal static path, which can be computed easily using Dijksta algorithm. Hence, the FPL algorithm works efficiently although the number of paths might be exponential and applying WM is unrealistic.

5 Idealized Settings

Every algorithm in the on-line MDP faces two major obstacles. The first is the computational issue since there is an exponential number of polices; the second is "state problem", the current state (distribution over the states) depends on the algorithm past and therefore the current reward depends on both the current action and the past actions, while the common assumption in *every* best expert algorithm is that the current reward is independent of the past. In this section we concentrate only on the first problem and define an idealized setting in which the second problem is irrelevant. In the idealized setting in *every* time step the algorithm chooses a policy and observes its average reward, $\eta_r(\pi)$. Therefore, the algorithm does not have a "state" anymore and the current reward is not effected by the previous decisions.

We start by describing and analyzing the naive approach. The naive algorithm uses a best expert algorithm satisfying Assumption 4.1, where each deterministic policy is an expert and the loss function for a policy π at time t is $\eta_{r_t}(\pi)$.

FPL in MDPs
for $t = 1$ *to* ∞ **do**
Choose p_t uniform at random in $\left[[0, |S||A|\sqrt{t}]^{|S||A|}\right]$;
Calculate the optimal policy, π_t for the MDP with reward function $r = \sum_{i=1}^{t} r_t + p_t$;
Use π_t at time $t + 1$;
end

Figure 1: FPL in Idealized setting

Since there is no state in the idealized setting we can use directly the standard best experts algorithms, with $|A|^{|S|}$ experts where the reward $\eta_r(\pi)$ is bounded by one. Substituting these quantities in the best expert performance guarantees we obtain the following Proposition.

Proposition 5.1 *Let A be the naive algorithm in the idealized settings. Then for all reward functions $r_1, r_2, \ldots r_T$ and for all stationary policies π,*

$$\frac{1}{T}\sum_{t=1}^{T} \eta_{r_t}(\pi_t) \geq \frac{1}{T}\sum_{t=1}^{T} \eta_{r_t}(\pi) - \sqrt{|S| \log |A|/T},$$

where π_t is the policy produced by A at time t. Also the naive algorithm has space complexity $O(|A|^{|S|})$ and in each time step it makes $O(|A|^{|S|})$ to calculate π_t.

The naive approach is clearly infeasible because of the space and time complexity. However, the MDP has a structure and therefore one can exploit them to give efficient algorithms. Algorithm 1 is the FPL algorithm adapted to MDPs.

The analysis of this algorithm in MDPs was also done independently by McMahan *et al.* [16].

Proposition 5.2 *For FPL in MDPs in the idealized settings, for all reward functions $r_1, r_2, \ldots r_T$ and for all stationary policies π,*

$$\frac{1}{T}\sum_{t=1}^{T} \eta_{r_t}(\pi_t) \geq \frac{1}{T}\sum_{t=1}^{T} \eta_{r_t}(\pi) - \sqrt{|S||A|/T},$$

where π_t is the policy produced by FPL at time t. Also the space complexity $O(|A||S|)$ and in each time step it runs in time polynomial in $|A|$ and $|S|$ to calculate π_t.

Proof: The FPL algorithm requires the cost function to be additive and its complexity is as the complexity of computing the optimal static problem. We first show that the cost

MDP-E
Put in every state a best expert algorithm B_s;
for $t = 1$ *to* ∞ **do**
Let $a_t(s)$ be the distribution over action of B_s at time t;
Let π_t be $\pi_t(s) = a_t(s)$;
Use policy π_t and obtain r_t from the environment;
Feed B_s with gain function $Q_{\pi_t, r_t}(s, \cdot)$
end

Figure 2: MDP Expert algorithm

function is additive, i.e., $\eta_R(\pi) = \sum_{i=1}^{t} \eta_{r_t}(\pi)$, where $R = \sum_{i=1}^{t} r_t$

$$\begin{aligned}
\sum_{i=1}^{t} \eta_{r_t}(\pi) &= \sum_{i=1}^{t} \sum_{s \in S} d_\pi(s)\pi(a|s)r(s,a) \\
&= \sum_{s \in S} \sum_{i=1}^{t} d_\pi(s)\pi(a|s)r(s,a) \\
&= \sum_{s \in S} d_\pi(s)\pi(a|s) \sum_{i=1}^{t} r(s,a) \\
&= \eta_R(\pi).
\end{aligned}$$

Next we would like to calculate the algorithm parameters, we know that the cost function is bounded by 1, we also have that for any two policies π, π', $\|d_\pi - d_{\pi'}\| \leq 2$, and that the number of the algorithm states is $|S||A|$. Substituting in Proposition 4.3 gives the desired bound. The space complexity is $|S||A|$ since we only need to store the cumulative reward for every state action pair. The time complexity of solving the static problem is that of solving an MDP and it can be done in time polynomial in $|S|$ and $|A|$. ∎

Next we present our algorithm which is as efficient as the FPL algorithm and its regret is slightly better. In contrary to FPL this algorithm is not general and it exploits directly the MDP structure. Later, we show that this algorithm translates to a good algorithm in the general setting as well. The algorithm is intuitive and simple to describe. The algorithm uses best expert algorithms which satisfy Assumption 4.1, but instead of using them in the policy level as in the naive approach it associates each state with a best expert algorithm where each expert corresponds to an action. The policy is defined by the product of all best expert algorithms distributions. The immediate question is what loss function should we feed to each expert. It turns out Q_{π_t, r_t} is appropriate. Note that although the best expert algorithm are "local" they contain "global" information by using Q_{π_t, r_t}.

Theorem 5.3 *For all sequences $r_1, r_2, \ldots r_T$, the MDP experts algorithm have the following*

performance bound. For all π,

$$\sum_{t=1}^{T} \eta_{r_t}(\pi_t) \geq \sum_{t=1}^{T} \eta_{r_t}(\pi) - \sqrt{3\tau T \log |A|}$$

where $\pi_1, \pi_2, \ldots \pi_T$ is the sequence of policies generated by $\mathcal{A}$ in response to $r_1, r_2, \ldots r_T$.

Before proving the Theorem, we provide a technical lemma, which is a variant of a result in Kakade [10].

Lemma 5.4 *For all policies π and π',*

$$\eta_r(\pi') - \eta_r(\pi) = \mathrm{E}_{s\sim d_{\pi'}}[Q_{\pi,r}(s,\pi') - Q_{\pi,r}(s,\pi)]$$

Proof: Note that by definition of stationarity, if the state distribution is at $d_{\pi'}$, then the next state distribution is also $d_{\pi'}$ if π' is followed. More formally, if $s \sim d_{\pi'}$, $a \sim \pi'(a|s)$, and $s' \sim P_{sa}$, then $s' \sim d_{\pi'}$. Using this and equation 1, we have:

$$\begin{aligned}
\mathrm{E}_{s\sim d_{\pi'}}[Q_{\pi,r}(s,\pi')] &= \mathrm{E}_{s\sim d_{\pi'},a\sim\pi'}[Q_{\pi,r}(s,a)] \\
&= \mathrm{E}_{s\sim d_{\pi'},a\sim\pi'}[r(s,a) - \eta_r(\pi) + \mathrm{E}_{s'\sim P_{sa}}[Q_{\pi,r}(s',\pi)] \\
&= \mathrm{E}_{s\sim d_{\pi'},a\sim\pi'}[r(s,a) - \eta_r(\pi)] + \mathrm{E}_{s\sim d_{\pi'}}[Q_{\pi,r}(s,\pi)] \\
&= \eta_r(\pi') - \eta_r(\pi) + \mathrm{E}_{s\sim d_{\pi'}}[Q_{\pi,r}(s,\pi)]
\end{aligned}$$

Rearranging terms leads to the result. ∎

The Lemma shows why our choice to feed each experts algorithm Q_{π_t,r_t} was appropriate. Now we complete the proof of the above theorem.

Proof: [Proof of Theorem 5.3]Using the assumed regret in assumption 4.1,

$$\begin{aligned}
\sum_{t=1}^{T} \eta_{r_t}(\pi) - \sum_{t=1}^{T} \eta_{r_t}(\pi_t) &= \sum_{t=1}^{T} \mathrm{E}_{s\sim d_\pi}[Q_{\pi_t,r_t}(s,\pi) - Q_{\pi_t,r_t}(s,\pi_t)] \\
&= \mathrm{E}_{s\sim d_\pi}[\sum_{t=1}^{T} Q_{\pi_t,r_t}(s,\pi) - Q_{\pi_t,r_t}(s,\pi_t)] \\
&\leq \mathrm{E}_{s\sim d_\pi}[\sqrt{3\tau T \log A}] \\
&= \sqrt{3\tau T \log A}
\end{aligned}$$

where we used the fact that d_π does not depend on the time in the second step. ∎

6 The General Setting

Now we provide our main result showing how to use any generic experts algorithm in the general setting, where the current reward is influenced by the previous policies.

While in the previous section we use only part of Assumption 4.1, as we shall see, it is important that this 'slow change' condition be satisfied in the general setting. Intuitively, our experts algorithms will be using a similar policy for significantly long periods of time.

Note that although we moved form the idealized settings to the general settings and our target function had changed our algorithm, Algorithm 2 remaind unchanged surprisingly. We now state our main theorem.

Theorem 6.1 *Let $\mathcal{A}$ be the MDP experts algorithm. Then for all reward functions $r_1, r_2, \ldots r_T$ and for all stationary policies π,*

$$V_{r_1,r_2,\ldots r_T}(\mathcal{A}) \geq V_{r_1,r_2,\ldots r_T}(\pi) - 4\tau^2\sqrt{\frac{\log|A|}{T}} - \sqrt{\frac{3\tau\log|A|}{T}} - \frac{4\tau}{T}$$

The regret goes to 0 at the rate $O(1/\sqrt{T})$, as is the case with experts algorithms, which do not have a state. Furthermore, the bound does *not depend* on the size of the state space.

6.1 The Analysis

The analysis is naturally divided into two parts. First, we use the performance bounds of the algorithm in the idealized setting. Then we take into account the slow change of the policies to show that the actual performance is similar to the instantaneous performance.

Taking Mixing Into Account: This subsection relates the values V to the sums of average reward used in the idealized setting.

Theorem 6.2 *For all sequences $r_1, r_2, \ldots r_T$ and for all $\mathcal{A}$*

$$|V_{r_1,r_2,\ldots r_T}(\mathcal{A}) - \frac{1}{T}\sum_{t=1}^{T}\eta_{r_t}(\pi_t)| \leq 4\tau^2\sqrt{\frac{\log|A|}{T}} + \frac{2\tau}{T}$$

where $\pi_1, \pi_2, \ldots \pi_T$ is the sequence of policies generated by $\mathcal{A}$ in response to $r_1, r_2, \ldots r_T$.

Since the above holds for all $\mathcal{A}$ (including those $\mathcal{A}$ which are the constant policy π), then combining this with Theorem 5.3 (once with $\mathcal{A}$ and once with π) completes the proof of Theorem 6.1. We now prove the above.

The following simple Lemma is useful. It shows how close are the next state distributions when following π_t rather than π_{t+1}.

Lemma 6.3 *Let π and π' be such that $\|\pi(\cdot|s) - \pi'(\cdot|s)\|_1 \leq \epsilon$. Then for any state distribution d, we have $\|dP^{\pi} - dP^{\pi'}\|_1 \leq \varepsilon$.*

Proof: Consider the case when d is a delta function on s. The difference in the next state distributions, $\|dP^{\pi} - dP^{\pi'}\|_1$, is:

$$\begin{aligned}\sum_{s'} |[P^{\pi}]_{s,s'} - [P^{\pi'}]_{s,s'}| &= \sum_{s'}\sum_{a} |P_{s,a}(s')(\pi(a|s) - \pi'(a|s))| \\ &\leq \sum_{s',a} P_{s,a}(s')|\pi(a|s) - \pi'(a|s)| \\ &= \sum_{a} |\pi(a|s) - \pi'(a|s)| \leq \varepsilon\end{aligned}$$

Linearity of expectation leads to the result for arbitrary d. ∎

Analogous to the definition of $d_{\pi,t}$, we define $d_{\mathcal{A},t}$

$$d_{\mathcal{A},t} = \Pr[s_t = s|d_1, \mathcal{A}]$$

which is the probability that the state at time t is s given that $\mathcal{A}$ has been followed.

Lemma 6.4 *Let $\pi_1, \pi_2, \ldots \pi_T$ be the sequence of policies generated by $\mathcal{A}$ in response to $r_1, r_2, \ldots r_T$. We have*

$$\|d_{\mathcal{A},t} - d_{\pi_t}\|_1 \leq 2\tau^2\sqrt{\frac{\log|A|}{t}} + 2e^{-t/\tau}$$

Proof: Let $k \leq t$. Using our experts assumption, it is straightforward to see that that the change in the policy over k steps is $|\pi_k(\cdot|s) - \pi_t(\cdot|s)|_1 \leq (t-k)\sqrt{\log|A|/t}$. Using this with $d_{\mathcal{A},k} = d_{\mathcal{A},k-1}P(\pi_k)$ and $d_{\pi_t}P^{\pi_t} = d_{\pi_t}$, we have

$$\begin{aligned}\|d_{\mathcal{A},k} - d_{\pi_t}\|_1 &= \|d_{\mathcal{A},k-1}P^{\pi_k} - d_{\pi_t}\|_1 \\ &\leq \|d_{\mathcal{A},k-1}P^{\pi_t} - d_{\pi_t}\|_1 + \|d_{\mathcal{A},k-1}P^{\pi_k} - d_{\mathcal{A},k-1}P^{\pi_t}\|_1 \\ &\leq \|d_{\mathcal{A},k-1}P^{\pi_t} - d_{\pi_t}P^{\pi_t}\|_1 + 2(t-k)\sqrt{\log|A|/t} \\ &\leq e^{-1/\tau}\|d_{\mathcal{A},k-1} - d_{\pi_t}\|_1 + 2(t-k)\sqrt{\log|A|/t}\end{aligned}$$

where we have used the last lemma in the third step and our contraction assumption 3.1 in the second to last step. Recursing on the above equation leads to:

$$\begin{aligned}\|d_{\mathcal{A},t} - d_{\pi_t}\| &\leq 2\sqrt{\log|A|/t}\sum_{k=t}^{2}(t-k)e^{-(t-k)/\tau} + e^{-t/\tau}\|d_1 - d_{\pi_t}\| \\ &\leq 2\sqrt{\log|A|/t}\sum_{k=1}^{\infty} ke^{-k/\tau} + 2e^{-t/\tau}\end{aligned}$$

The sum is bounded by an integral from 0 to ∞, which evaluates to τ^2. ∎

We are now ready to prove the mixing theorem.

Proof:[Proof of Theorem 6.2] By definition of V,

$$\begin{aligned} V_{r_1,r_2,\ldots r_T}(\mathcal{A}) &= \frac{1}{T}\sum_{t=1}^{T} E_{s\sim d_{\mathcal{A},t},a\sim\pi_t}[r_t(s,a)] \\ &\leq \frac{1}{T}\sum_{t=1}^{T} E_{s\sim d_{\pi_t},a\sim\pi_t}[r_t(s,a)] + \frac{1}{T}\sum_{t=1}^{T}\|d_{\mathcal{A},t} - d_{\pi_t}\|_1 \\ &\leq \frac{1}{T}\sum_{t=1}^{T}\eta_{r_t}(\pi_t) + \frac{1}{T}\sum_{t=1}^{T}\left(2\tau^2\sqrt{\frac{\log|A|}{t}} + 2e^{-t/\tau}\right) \\ &\leq \frac{1}{T}\sum_{t=1}^{T}\eta_{r_t}(\pi_t) + 4\tau^2\sqrt{\frac{\log|A|}{T}} + \frac{2\tau}{T} \end{aligned}$$

where we have bounded the sums by integration in the second to last step. A symmetric argument leads to the result.

∎

Proof:[Proof of Theorem 6.1] We first relate $V_{r_1,r_2,\ldots r_T}(\pi)$ to $\sum_{t=1}^{T}\eta_{r_t}(\pi)$.

$$\begin{aligned} V_{r_1,r_2,\ldots r_T}(\pi) &= \frac{1}{T}\sum_{t=1}^{T} E_{s\sim d_{\pi,t},a\sim\pi}[r_t(s,a)] \\ &\leq \frac{1}{T}\sum_{t=1}^{T} E_{s\sim d_{\pi},a\sim\pi}[r_t(s,a)] + \frac{1}{T}\sum_{t=1}^{T}\|d_{\pi,t} - d_{\pi}\|_1 \\ &\leq \frac{1}{T}\sum_{t=1}^{T}\eta_{r_t}(\pi) + \frac{1}{T}\sum_{t=1}^{T}2e^{-t/\tau} \\ &\leq \frac{1}{T}\sum_{t=1}^{T}\eta_{r_t}(\pi) + \frac{2\tau}{T} \end{aligned}$$

Using this inequality and the former Theorem we are ready to complete our proof.

$$\begin{aligned} V_{r_1,r_2,\ldots r_T}(\pi) - V_{r_1,r_2,\ldots r_T}(\mathcal{A}) &\leq \frac{1}{T}\sum_{t=1}^{T}\eta_{r_t}(\pi) + \frac{2\tau}{T} - V_{r_1,r_2,\ldots r_T}(\mathcal{A}) \\ &\leq \frac{1}{T}\sum_{t=1}^{T}\eta_{r_t}(\pi) + \frac{2\tau}{T} - \left(\frac{1}{T}\sum_{t=1}^{T}\eta_{r_t}(\pi_t) - 4\tau^2\sqrt{\frac{\log|A|}{T}} - \frac{2\tau}{T}\right) \\ &= \frac{1}{T}\sum_{t=1}^{T}\eta_{r_t}(\pi) - \frac{1}{T}\sum_{t=1}^{T}\eta_{r_t}(\pi_t) + 4\tau^2\sqrt{\frac{\log|A|}{T}} + \frac{4\tau}{T} \\ &\leq 4\tau^2\sqrt{\frac{\log|A|}{T}} + \frac{4\tau}{T} + \sqrt{\frac{3\tau\log A}{T}}, \end{aligned}$$

where the first inequity is due to the former bound, the second is due to Theorem 6.2, and the last inequity is due to Theorem 5.3. ∎

7 Conclusions and Open Problems

This paper was a step into bridging reinforcement learning and adversarial on-line learning. We presented an efficient low regret algorithm for an on-line MDP. This extension over the standard MDP literature [19] is important since the Markovian assumption implied by the MDP is in many times relaxation of the true non-markovian world. An open problem which remains is the relaxation of the full information assumption into the bandit settings, where one can observe only its reward for the current state action.

8 Acknowledgments.

This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778, and by a grant from the Israel Science Foundation and an IBM faculty award. This publication only reflects the authors' views. The work was done while E.E was a graduate student in Tel Aviv University.

9 Bibliography.

References

[1] P. Auer, N. Cesa-Bianchi and C. Gentile, *Adaptive and Self-Confident On-Line Learning Algorithms*, Journal of Computer System Sciences, Vol. 64, pages 48–75, 2002

[2] A. Blum and A. Kalai, *Universal Portfolios With and Without Transaction Costs*, Machine Learning, 35:193-205, 1999.

[3] Dimitri P. Bertsekas and John N. Tsitsiklis, *Neuro-Dynamic Programming*, Athena Scientific, Belmont, MA, 1996

[4] Nicolò Cesa-Bianchi, Yoav Freund, David Haussler, David P. Helmbold, Robert E. Schapire, and Manfred K. Warmuth. *How to Use Expert Advice*, Journal of the ACM, Vol 44(3): 427-485, 1997.

[5] E. Even-Dar, Sham M. Kakade and Y. Mansour, *Experts in a Markov Decision Process*, Proceedings of Advances in Neural Information Processing Systems 17 (NIPS), 401-408, 2004

[6] D. de Farias and N. Megiddo, *How to Combine Expert (or Novice) Advice when Actions Impact the Environment*,Proceedings of Neural Information Processing Systems (NIPS 2003).

[7] D. de Farias and N. Megiddo, *Exploration-Exploitation Tradeoffs for Experts Algorithms in Reactive Environments*, Proceedings of Neural Information Processing Systems (NIPS 2004).

[8] J. Hannan, *Approximation to Bayes Risk in Repeated Play*, Contribution to The Theory of Games, III, editors M. Dresher and A. W. Tucker and P. Wolde, Princeton University Press, 97 -139, 1957

[9] D. Helmbold, R. Schapire, Y. Singer, and M. Warmuth *On-Line Portfolio Selection Using Multiplicative Updates* Mathemtical Finance, Vol 8, N0. 4, 325-347.

[10] Sham M. Kakade, *On the Sample Complexity of Reinforcement Learning*, University College London, 2003

[11] A. Kalai and S. Vempala, *Efficient Algorithms for On-line Optimization*, Journal of Computer and System Sciences 71(3): 291-307, 2005

[12] M. Kearns and S. Singh, *Near-Optimal Reinforcement Learning in Polynomial Time*, Machine Learning , 49(2-3): 209-232, 2002.

[13] J. Kivinen and M. Warmuth *Additive Versus Exponentiated Gradient Updates for Linear Prediction*, in Journal of Information and Computation, vol. 132, no. 1, pp. 1-64, 1997

[14] N. Littlestone and M. K. Warmuth", *The weighted majority algorithm*, Information and Computation, volume 108(2), pages 212–261, 1994

[15] H. McMahan and G. Gordon and A. Blum, *Planning in the Presence of Cost Functions Controlled by an Adversary*, Proceedings of 20th International Conference on Machine Learning (ICML), 536-543, 2003

[16] H. McMahan and G. Gordon and A. Blum, *Personal Communication*, 2003

[17] A. Nilim and L. El Ghaoui, *Robust Solutions to Markov Decision Problems with Uncertain Transition Matrices*, Operations Resaerch, Vol 53, pages 780-798, 2005

[18] M. Puterman, *Markov Decision Processes*, Wiley-Interscience, 1994

[19] R. Sutton and A. Barto, *Reinforcement Learning. An Introduction.*, MIT Press, Cambridge, MA, 1998