Technical Report IDSIA-08-08

On the Possibility of Learning in Reactive Environments with Arbitrary Dependence

Daniil Ryabko

IDSIA, Galleria 2, CH-6928
Manno-Lugano, Switzerland*
daniil@idsia.ch

Marcus Hutter

RSISE @ ANU and SML @ NICTA
Canberra, ACT, 0200, Australia
http://www.hutter1.net

October 2008

Abstract

We address the problem of reinforcement learning in which observations may exhibit an arbitrary form of stochastic dependence on past observations and actions, i.e. environments more general than (PO)MDPs. The task for an agent is to attain the best possible asymptotic reward where the true generating environment is unknown but belongs to a known countable family of environments. We find some sufficient conditions on the class of environments under which an agent exists which attains the best asymptotic reward for any environment in the class. We analyze how tight these conditions are and how they relate to different probabilistic assumptions known in reinforcement learning and related fields, such as Markov Decision Processes and mixing conditions.

Contents

Keywords

Reinforcement learning, asymptotic average value, self-optimizing policies, (non) Markov decision processes.

*This work was supported by the Swiss NSF grants 200020-107616 and 200021-113364.

1 Introduction

Many real-world "learning" problems (like learning to drive a car or playing a game) can be modelled as an agent π that interacts with an environment μ and is (occasionally) rewarded for its behavior. We are interested in agents which perform well in the sense of having high long-term reward, also called the value $V(\mu,\pi)$ of agent π in environment μ. If μ is known, it is a pure (non-learning) computational problem to determine the optimal agent $\pi^\mu := \arg\max_\pi V(\mu,\pi)$. It is far less clear what an "optimal" agent means, if μ is unknown. A reasonable objective is to have a single policy π with high value simultaneously in many environments. We will formalize and call this criterion *self-optimizing* later.

Learning approaches in reactive worlds. Reinforcement learning, sequential decision theory, adaptive control theory, and active expert advice, are theories dealing with this problem. They overlap but have different core focus: Reinforcement learning algorithms [SB98] are developed to learn μ or directly its value. Temporal difference learning is computationally very efficient, but has slow asymptotic guarantees (only) in (effectively) small observable MDPs. Others have faster guarantee in finite state MDPs [BT99]. There are algorithms [EDKM05] which are optimal for any finite connected POMDP, and this is apparently the largest class of environments considered. In sequential decision theory, a Bayes-optimal agent π^* that maximizes $V(\xi,\pi)$ is considered, where ξ is a mixture of environments $\nu \in \mathcal{C}$ and $\mathcal{C}$ is a class of environments that contains the true environment $\mu \in \mathcal{C}$ [Hut05]. Policy π^* is self-optimizing in an arbitrary (e.g. non-POMDP) class $\mathcal{C}$, provided $\mathcal{C}$ allows for self-optimizingness [Hut02]. Adaptive control theory [KV86] considers very simple (from an AI perspective) or special systems (e.g. linear with quadratic loss function), which sometimes allow computationally and data efficient solutions. Action with expert advice [dFM04, PH05, PH06, CBL06] constructs an agent (called master) that performs nearly as well as the best agent (best expert in hindsight) from some class of experts, in *any* environment ν. The important special case of passive sequence prediction in arbitrary unknown environments, where the actions=predictions do not affect the environment is comparably easy [Hut03, HP04].

The difficulty in active learning problems can be identified (at least, for countable classes) with *traps* in the environments. Initially the agent does not know μ, so has asymptotically to be forgiven in taking initial "wrong" actions. A well-studied such class are ergodic MDPs which guarantee that, from any action history, every state can be (re)visited [Hut02].

What's new. The aim of this paper is to characterize as general as possible classes $\mathcal{C}$ in which self-optimizing behaviour is possible, more general than POMDPs. To do this we need to characterize classes of environments that forgive. For instance, exact state recovery is unnecessarily strong; it is sufficient being able to recover high rewards, from whatever states. Further, in many real world problems there is no information available about the "states" of the environment (e.g. in POMDPs) or the environment may exhibit long history dependencies.

Rather than trying to model an environment (e.g. by MDP) we try to identify the conditions sufficient for learning. Towards this aim, we propose to consider only environments in which, after any arbitrary finite sequence of actions, the best value is still achievable. The performance criterion here is asymptotic average reward. Thus we consider such environments for which there exists a policy whose asymptotic average reward exists and upper-bounds asymptotic average reward of any other policy. Moreover, the same property should hold after any finite sequence of actions has been taken (no traps). We call such environments *recoverable.* If we only want to get ε-close to the optimal value infinitely often with decreasing ε (that is, to have the same upper limit for the average value), then this property is already sufficient.

Yet recoverability in itself is not sufficient for identifying behaviour which results in optimal limiting average value. We require further that, from any sequence of k actions, it is possible to return to the optimal level of reward in $o(k)$ steps; that is, it is not just possible to recover after any sequence of (wrong) actions, but it is possible to recover fast. Environments which possess this property are called *value-stable.* (These conditions will be formulated in a probabilistic form.)

We show that for any countable class of value-stable environments there exists a policy which achieves the best possible value in any of the environments from the class (i.e. is *self-optimizing* for this class).

Furthermore, we present some examples of environments which possess value-stability and/or recoverability. In particular, any ergodic MDP can be easily shown to be value-stable. A mixing-type condition which implies value-stability is also demonstrated. In addition, we provide a construction allowing to build examples of value-stable and/or recoverable environments which are not isomorphic to a finite POMDP, thus demonstrating that the class of value-stable environments is quite general.

Finally, we consider environments which are not recoverable but still are value-stable. In other words, we consider the question of what it means to be optimal in an environment which does not "forgive" wrong actions. Even in such cases some policies are better than others, and we identify some conditions which are sufficient for learning a policy that is optimal from some point on.

It is important in our argument that the class of environments for which we seek a self-optimizing policy is countable, although the class of all value-stable environments is uncountable. To find a set of conditions necessary and sufficient for learning which do not rely on countability of the class is yet an open problem. However, from a computational perspective countable classes are sufficiently large (e.g. the class of all computable probability measures is countable).

Contents. The paper is organized as follows. Section 2 introduces necessary notation of the agent framework. In Section 3 we define and explain the notion of value-stability, which is central to the paper, and a weaker but simpler notion of recoverability. Section 4 presents the theorems about self-optimizing policies for classes of value-stable environments and recoverable environments. In Section 5 we discuss what can be achieved if the environments are not recoverable. Section 6 illustrates the applicability of the theorems by providing examples of value-stable and recoverable environments. In Section 7 we discuss necessity of the conditions of the main theorems. Section 8 provides some discussion of the results and an outlook to future research. Formal proofs of the main theorems are given in Section A, while Sections 4 and 5 contain only intuitive explanations.

2 Notation and Definitions

We essentially follow the notation of [Hut02, Hut05].

Strings and probabilities. We use letters $i,k,l,m,n \in I\!N$ for natural numbers, and denote the cardinality of sets $\mathcal{S}$ by $\#\mathcal{S}$. We write $\mathcal{X}^*$ for the set of finite strings over some alphabet $\mathcal{X}$, and $\mathcal{X}^\infty$ for the set of infinite sequences. For a string $x \in \mathcal{X}^*$ of length $\ell(x) = n$ we write $x_1x_2...x_n$ with $x_t \in \mathcal{X}$ and further abbreviate $x_{k:n} := x_kx_{k+1}...x_{n-1}x_n$ and $x_{<n} := x_1...x_{n-1}$. Finally, we define $x_{k..n} := x_k + ... + x_n$, provided elements of $\mathcal{X}$ can be added.

We assume that sequence $\omega = \omega_{1:\infty} \in \mathcal{X}^\infty$ is sampled from the "true" probability measure μ, i.e. $\mathbf{P}[\omega_{1:n} = x_{1:n}] = \mu(x_{1:n})$. We denote expectations w.r.t. μ by $\mathbf{E}$, i.e. for a function $f : \mathcal{X}^n \to I\!R$, $\mathbf{E}[f] = \mathbf{E}[f(\omega_{1:n})] = \sum_{x_{1:n}} \mu(x_{1:n}) f(x_{1:n})$. When we use probabilities and expectations with respect to other measures we make the notation explicit, e.g. $\mathbf{E}_\nu$ is the expectation with respect to ν. Measures ν_1 and ν_2 are called *singular* if there exists a set A such that $\nu_1(A) = 0$ and $\nu_2(A) = 1$.

The agent framework is general enough to allow modelling nearly any kind of (intelligent) system [RN95]. In cycle k, an agent performs *action* $y_k \in \mathcal{Y}$ (output) which results in *observation* $o_k \in \mathcal{O}$ and *reward* $r_k \in \mathcal{R}$, followed by cycle $k+1$ and so on. We assume that the action space $\mathcal{Y}$, the observation space $\mathcal{O}$, and the reward space $\mathcal{R} \subset I\!R$ are finite, w.l.g. $\mathcal{R} = \{0,...,r_{max}\}$. We abbreviate $z_k := y_kr_ko_k \in \mathcal{Z} := \mathcal{Y} \times \mathcal{R} \times \mathcal{O}$ and $x_k = r_ko_k \in \mathcal{X} := \mathcal{R} \times \mathcal{O}$. An agent is identified with a (probabilistic) *policy* π. Given *history* $z_{<k}$, the probability that agent π acts y_k in cycle k is (by definition) $\pi(y_k|z_{<k})$. Thereafter, *environment* μ provides (probabilistic) reward r_k and observation o_k, i.e. the probability that the agent perceives x_k is (by definition) $\mu(x_k|z_{<k}y_k)$. Note that the policy and the environment are allowed to depend on the complete history. We do not make any MDP or POMDP assumption here, and we don't talk about states of the environment, only about observations. Each (policy,environment) pair (π,μ) generates an I/O sequence $z_1^{\pi\mu} z_2^{\pi\mu}....$ Mathematically, the history $z_{1:k}^{\pi\mu}$ is a random variable with probability

$$\mathbf{P}\left(z_{1:k}^{\pi\mu} = z_{1:k}\right) \;=\; \pi(y_1) \cdot \mu(x_1|y_1) \cdot ... \cdot \pi(y_k|z_{<k}) \cdot \mu(x_k|z_{<k}y_k).$$

Since value maximizing policies can always be chosen deterministic, there is no real need to consider probabilistic policies, and henceforth we consider deterministic policies p. We assume that $\mu \in \mathcal{C}$ is the true, but unknown, environment, and $\nu \in \mathcal{C}$ a generic environment.

3 Setup

For an environment ν and a policy p define random variables (upper and lower average value)

$$\overline{V}(\nu,p) \;:=\; \limsup_m \left\{\tfrac{1}{m} r^{p\nu}_{1..m}\right\} \quad\text{and}\quad \underline{V}(\nu,p) \;:=\; \liminf_m \left\{\tfrac{1}{m} r^{p\nu}_{1..m}\right\}$$

where $r_{1..m} := r_1 + ... + r_m$. If there exists a constant $\overline{V}$ or a constant $\underline{V}$ such that

$$\overline{V}(\nu,p) \;=\; \overline{V} \text{ a.s., or } \underline{V}(\nu,p) \;=\; \underline{V} \text{ a.s.}$$

then we say that the upper limiting average or (respectively) lower average value exists, and denote it by $\overline{V}(\nu,p) := \overline{V}$ (or $\underline{V}(\nu,p) := \underline{V}$). If both upper and lower average limiting values exist and are equal then we simply say that average limiting value exist and denote it by $V(\nu,p) := \overline{V}(\nu,p) = \underline{V}(\nu,p)$

An environment ν is *explorable* if there exists a policy p_ν such that $V(\nu,p_\nu)$ exists and $\overline{V}(\nu,p) \le V(\nu,p_\nu)$ with probability 1 for every policy p. In this case define $V^*_\nu := V(\nu,p_\nu)$. An environment ν is *upper explorable* if there exists a policy p_ν such that $\overline{V}(\nu,p_\nu)$ exists and $\overline{V}(\nu,p) \le \overline{V}(\nu,p_\nu)$ with probability 1 for every policy p. In this case define $\overline{V}^*_\nu := \overline{V}(\nu,p_\nu)$.

A policy p is *self-optimizing* for a set of explorable environments $\mathcal{C}$ if $V(\nu,p) = V^*_\nu$ for every $\nu \in \mathcal{C}$. A policy p is *upper self-optimizing* for a set of explorable environments $\mathcal{C}$ if $\overline{V}(\nu,p) = \overline{V}^*_\nu$ for every $\nu \in \mathcal{C}$.

In the case when we we wish to obtain the optimal average value for any environment in the class we will speak about self-optimizing policies, whereas if we are only interested in obtaining the upper limit of the average value then we will speak about upper self-optimizing policies. It turns out that the latter case is much simpler. The next two definitions present conditions on the environments which will be shown to be sufficient to achieve the two respective goals.

Definition 1 (recoverable)**.** *We call an upper explorable environment ν recoverable if for any history $z_{<k}$ such that $\nu(x_{<k}|y_{<k}) > 0$ there exists a policy p such that*

$$\mathbf{P}(\overline{V}(\nu,p) = \overline{V}^* | z_{<k}) = 1.$$

Conditioning on the history $z_{<k}$ means that we take ν-conditional probabilities (conditional on $x_{<k}$) and first $k-1$ actions of the policy p are replaced by $y_{<k}$.

Recoverability means that after taking any finite sequence of (possibly sub-optimal) actions it is still possible to obtain the same upper limiting average value as an optimal policy would obtain. The next definition is somewhat more complex.

Definition 2 (value-stable environments)**.** *An explorable environment ν is* value-stable *if there exist a sequence of numbers $r^\nu_i \in [0,r_{max}]$ and two functions $d_\nu(k,\varepsilon)$ and $\varphi_\nu(n,\varepsilon)$ such that $\frac{1}{n} r^\nu_{1..n} \to V^*_\nu$, $d_\nu(k,\varepsilon) = o(k)$, $\sum_{n=1}^\infty \varphi_\nu(n,\varepsilon) < \infty$ for every fixed ε, and for every k and every history $z_{<k}$ there exists a policy $p = p^{z_{<k}}_\nu$ such that*

$$\mathbf{P}\left(r^\nu_{k..k+n} - r^{p\nu}_{k..k+n} > d_\nu(k,\varepsilon) + n\varepsilon \mid z_{<k}\right) \le \varphi_\nu(n,\varepsilon). \tag{1}$$

First of all, this condition means that the strong law of large numbers for rewards holds uniformly over histories $z_{<k}$; the numbers r_i^ν here can be thought of as expected rewards of an optimal policy. Furthermore, the environment is "forgiving" in the following sense: from any (bad) sequence of k actions it is possible (knowing the environment) to recover up to $o(k)$ reward loss; to recover means to reach the level of reward obtained by the optimal policy which from the beginning was taking only optimal actions. That is, suppose that a person A has made k possibly suboptimal actions and after that "realized" what the true environment was and how to act optimally in it. Suppose that a person B was from the beginning taking only optimal actions. We want to compare the performance of A and B on first n steps after the step k. An environment is value stable if A can catch up with B except for $o(k)$ gain. The numbers r_i^ν can be thought of as expected rewards of B; A can catch up with B up to the reward loss $d_\nu(k,\varepsilon)$ with probability $\varphi_\nu(n,\varepsilon)$, where the latter does not depend on past actions and observations (the law of large numbers holds uniformly).

Examples of value-stable environments will be considered in Section 6.

4 Main Results

In this section we present the main self-optimizingness result along with an informal explanation of its proof, and a result on upper self-optimizingness, which turns out to have much more simple conditions.

Theorem 3 (value-stable⇒self-optimizing). *For any countable class $\mathcal{C}$ of value-stable environments, there exists a policy which is self-optimizing for $\mathcal{C}$.*

A formal proof is given in the appendix; here we give some intuitive justification. Suppose that all environments in $\mathcal{C}$ are deterministic. We will construct a self-optimizing policy p as follows: Let ν^t be the first environment in $\mathcal{C}$. The algorithm assumes that the true environment is ν^t and tries to get ε-close to its optimal value for some (small) ε. This is called an exploitation part. If it succeeds, it does some exploration as follows. It picks the first environment ν^e which has higher average asymptotic value than ν^t ($V^*_{\nu^e} > V^*_{\nu^t}$) and tries to get ε-close to this value acting optimally under ν^e. If it cannot get close to the ν^e-optimal value then ν^e is not the true environment, and the next environment can be picked for exploration (here we call "exploration" successive attempts to exploit an environment which differs from the current hypothesis about the true environment and has a higher average reward). If it can, then it switches to exploitation of ν^t, exploits it until it is ε'-close to $V^*_{\nu^t}$, $\varepsilon' < \varepsilon$ and switches to ν^e again this time trying to get ε'-close to V_{ν^e}; and so on. This can happen only a finite number of times if the true environment is ν^t, since $V^*_{\nu^t} < V^*_{\nu^e}$. Thus after exploration either ν^t or ν^e is found to be inconsistent with the current history. If it is ν^e then just the next environment ν^e such that $V^*_{\nu^e} > V^*_{\nu^t}$ is picked for exploration. If it is ν^t then the first consistent environment is picked for exploitation (and denoted ν^t). This in turn can happen only a finite number of times before the true environment ν is picked as ν^t. After this, the algorithm still continues its exploration attempts, but can always keep within $\varepsilon_k \to 0$ of the optimal value. This is ensured by $d(k)=o(k)$.

The probabilistic case is somewhat more complicated since we can not say whether an environment is "consistent" with the current history. Instead we test each environment for consistency as follows. Let ξ be a mixture of all environments in $\mathcal{C}$. Observe that together with some fixed policy each environment μ can be considered as a measure on $\mathcal{Z}^\infty$. Moreover, it can be shown that (for any fixed policy) the ratio $\frac{\nu(z_{<n})}{\xi(z_{<n})}$ is bounded away from zero if ν is the true environment μ and tends to zero if ν is singular with μ (in fact, here singularity is a probabilistic analogue of inconsistency). The exploration part of the algorithm ensures that at least one of the environments ν^t and ν^e is singular with ν on the current history, and a succession of tests $\frac{\nu(z_{<n})}{\xi(z_{<n})} \geq \alpha_s$ with $\alpha_s \to 0$ is used to exclude such environments from consideration.

Upper self-optimizingness. Next we consider the task in which our goal is more moderate. Rather than trying to find a policy which will obtain the same average limiting value as an optimal one for any environment in a certain class, we will try to obtain only the optimum upper limiting average. That is, we will try to find a policy which infinitely often gets as close as desirable to the maximum possible average value. It turns out that in this case a much simpler condition is sufficient: recoverability instead of value-stability.

Theorem 4 (recoverable⇒upper self-optimizing)**.** *For any countable class $\mathcal{C}$ of recoverable environments, there exists a policy which is upper self-optimizing for $\mathcal{C}$.*

A formal proof can be found in Section A; its idea is as follows. The upper self-optimizing policy p to be constructed will loop through all environments in $\mathcal{C}$ in such a way that each environment is tried infinitely often, and for each environment the agent will try to get ε-close (with decreasing ε) to the upper-limiting average value, until it either manages to do so, or a special stopping condition holds: $\frac{\nu(z_{<n})}{\xi(z_{<n})} < \alpha_s$, where α_s is decreasing accordingly. This condition necessarily breaks if the upper limiting average value cannot be achieved.

5 Non-recoverable environments

Before proceeding with examples of value-stable environments, we briefly discuss what can be achieved if an environment does not forgive initial wrong actions, that is, is not recoverable. It turns out that value-stability can be defined for non-recoverable environments as well, and optimal — in a worst-case sense — policies can be identified.

For an environment ν, a policy p and a history $z_{<k}$ such that $\nu(x_{<k}|y_{<k})>0$, if there exists a constant $\overline{V}$ or a constant $\underline{V}$ such that

$$P(\overline{V}(\nu,p) = \overline{V}|z_{<k})=1, \text{ or } P(\underline{V}(\nu,p) = \underline{V}|z_{<k})=1,$$

then we say that the upper conditional (on $z_{<k}$) limiting average or (respectively) lower conditional average value exists, and denote it by $\overline{V}(\nu,p,z_{<k}):=\overline{V}$ (or $\underline{V}(\nu,p,z_{<k}):=\underline{V}$). If both upper and lower conditional average limiting values exist and are equal then we say that that average conditional value exist and denote it by $V(\nu,p,(z_{<k})):=\overline{V}(\nu,p,z_{<k})=\underline{V}(\nu,p,z_{<k})$

Call an environment ν *strongly (upper) explorable* if for any history $z_{<k}$ such that $\nu(x_{<k}|y_{<k})>0$ there exists a policy $p_\nu^{z_{<k}}$ such that $V(\nu,p_\nu^{z_{<k}})$ ($\overline{V}(\nu,p_\nu^{z_{<k}})$) exists and $\overline{V}(\nu,p,z_{<k})\leq V(\nu,p_\nu^{z_{<k}},z_{<k})$ (respectively $\overline{V}(\nu,p,z_{<k})\leq\overline{V}(\nu,p_\nu^{z_{<k}},z_{<k})$) with probability 1 for every policy p. In this case define $V^*_\nu(z_{<k}):=V(\nu,p_\nu^{z_{<k}})$ (respectively $\overline{V}^*_\nu(z_{<k}):=\overline{V}(\nu,p_\nu^{z_{<k}})$).

For a strongly explorable environment ν define the worst-case optimal value

$$W^*_\nu:=\inf_{k,z_{<k}:\nu(x_{<k}>0)} V^*_\nu(z_{<k}),$$

and for a strongly upper explorable ν define the worst-case upper optimal value

$$\overline{W}^*_\nu:=\inf_{k,z_{<k}:\nu(x_{<k}>0)} \overline{V}^*_\nu(z_{<k}).$$

In words, the worst-case optimal value is the asymptotic average reward which is attainable with certainty after any finite sequence of actions has been taken.

Note that a recoverable explorable environment is also strongly explorable.

A policy p will be called *worst-case self-optimizing* or *worst-case upper self-optimizing* for a class of environments $\mathcal{C}$ if $\liminf\frac{1}{m}r^{p\nu}_{1..m}\geq W^*_\nu$, or (respectively) $\limsup\frac{1}{m}r^{p\nu}_{1..m}\geq\overline{W}^*_\nu$ with probability 1 for every $\nu\in\mathcal{C}$.

Definition 5 (worst-case value-stable environments)**.** *A strongly explorable environment ν is* worst-case value-stable *if there exists a sequence of numbers $r_i^\nu\in[0,r_{max}]$ and two functions $d_\nu(k,\varepsilon)$ and $\varphi_\nu(n,\varepsilon)$ such that $\frac{1}{n}r^\nu_{1..n}\to W^*_\nu$, $d_\nu(k,\varepsilon)=o(k)$, $\sum_{n=1}^\infty\varphi_\nu(n,\varepsilon)<\infty$ for every fixed ε, and for every k and every history $z_{<k}$ there exists a policy $p=p_\nu^{z_{<k}}$ such that*

$$\mathbf{P}\left(r^\nu_{k..k+n}-r^{p\nu}_{k..k+n}>d_\nu(k,\varepsilon)+n\varepsilon\mid z_{<k}\right)\leq\varphi_\nu(n,\varepsilon). \tag{2}$$

Note that a recoverable environment is value-stable if and only if it is worst-case value-stable.

Worst-case value stability helps to distinguish between irreversible actions (or "traps") and actions which result only in a temporary loss in performance; moreover, worst-case value-stability means that a temporary loss in performance can only be short (sublinear).

Finally, we can establish the following result (cf. Theorems 3 and 4).

Theorem 6 (worst-case self-optimizing). *(i) For any countable set of worst-case value-stable environments $\mathcal{C}$ there exist a policy p which is worst-case self-optimizing for $\mathcal{C}$.*

(ii) For any countable set of strongly upper explorable environments $\mathcal{C}$ there exist a policy p which is worst-case upper self-optimizing for $\mathcal{C}$.

The proof of this theorem is analogous to the proofs of Theorems 3 and 4; the differences are explained in Section A.

6 Examples

In this section we illustrate the results of the previous section with examples of classes of value-stable environments. These are also examples of recoverable environments, since recoverability is strictly weaker than value-stability. In the end of the section we also give some simple examples of recoverable but not value-stable environments.

We first note that passive environments are value-stable. An environment is called *passive* if the observations and rewards do not depend on the actions of the agent. Sequence prediction task provides a well-studied (and perhaps the only reasonable) class of passive environments: in this task the agent is required to give the probability distribution of the next observation given the previous observations. The true distribution of observations depends only on the previous observations (and does not depend on actions and rewards). Since we have confined ourselves to considering finite action spaces, the agent is required to give ranges of probabilities for the next observation, where the ranges are fixed beforehand. The reward 1 is given if all the ranges are correct and the reward 0 is given otherwise. It is easy to check that any such environment is value-stable with $r_i^\nu \equiv 1$, $d(k,\varepsilon) \equiv 1$, $\varphi(n,\varepsilon) \equiv 0$, since, knowing the distribution, one can always start giving the correct probability ranges (this defines the policy p_ν).

Obviously, there are active value stable environments too. The next proposition provides some conditions on mixing rates which are sufficient for value-stability; we do not intend to provide sharp conditions on mixing rates but rather to illustrate the relation of value-stability with mixing conditions.

We say that a stochastic process h_k, $k \in I\!N$ satisfies strong α-mixing conditions with coefficients $\alpha(k)$ if (see e.g. [Bos96])

$$\sup_{n \in I\!N} \sup_{B \in \sigma(h_1,\dots,h_n), C \in \sigma(h_{n+k},\dots)} |\, \mathbf{P}(B \cap C) - \mathbf{P}(B)\, \mathbf{P}(C)| \leq \alpha(k),$$

where $\sigma()$ stands for the sigma-algebra generated by the random variables in brackets. Loosely speaking, mixing coefficients α reflect the speed with which the process "forgets" about its past.

Proposition 7 (mixing and value-stability)**.** *Suppose that an explorable environment ν is such that there exist a sequence of numbers r_i^ν and a function $d(k)$ such that $\frac{1}{n}r_{1..n}^\nu \to V_\nu^*$, $d(k)=o(k)$, and for each $z_{<k}$ there exists a policy p such that the sequence $r_i^{p\nu}$ satisfies strong α-mixing conditions with coefficients $\alpha(k)=\frac{1}{k^{1+\varepsilon}}$ for some $\varepsilon>0$ and*

$$r_{k..k+n}^\nu - \mathbf{E}\left(r_{k..k+n}^{p\nu} \mid z_{<k}\right) \leq d(k)$$

for any n. Then ν is value-stable.

Proof. Using the union bound we obtain

$$\begin{aligned}
&\mathbf{P}\left(r_{k..k+n}^\nu - r_{k..k+n}^{p\nu} > d(k) + n\varepsilon\right) \\
&\leq I\left(r_{k..k+n}^\nu - \mathbf{E}\, r_{k..k+n}^{p\nu} > d(k)\right) + \mathbf{P}\left(\left|r_{k..k+n}^{p\nu} - \mathbf{E}\, r_{k..k+n}^{p\nu}\right| > n\varepsilon\right).
\end{aligned}$$

The first term equals 0 by assumption and the second term for each ε can be shown to be summable using [Bos96, Thm.1.3]: for a sequence of uniformly bounded zero-mean random variables r_i satisfying strong α-mixing conditions the following bound holds true for any integer $q\in[1,n/2]$

$$\mathbf{P}\left(|r_{1..n}| > n\varepsilon\right) \leq ce^{-\varepsilon^2 q/c} + cq\alpha\left(\frac{n}{2q}\right)$$

for some constant c; in our case we just set $q=n^{\frac{\varepsilon}{2+\varepsilon}}$. □

(PO)MDPs. Applicability of Theorem 3 and Proposition 7 can be illustrated on (PO)MDPs. We note that self-optimizing policies for (uncountable) classes of finite ergodic MDPs and POMDPs are known [BT99, EDKM05]; the aim of the present section is to show that value-stability is a weaker requirement than the requirements of these models, and also to illustrate applicability of our results. We call μ a (stationary) *Markov decision process* (MDP) if the probability of perceiving $x_k\in\mathcal{X}$, given history $z_{<k}y_k$ only depends on $y_k\in\mathcal{Y}$ and x_{k-1}. In this case $x_k\in\mathcal{X}$ is called a *state*, $\mathcal{X}$ the *state space*. An MDP μ is called *ergodic* if there exists a policy under which every state is visited infinitely often with probability 1. An MDP with a stationary policy forms a Markov chain.

An environment is called a (finite) *partially observable MDP* (POMDP) if there is a sequence of random variables s_k taking values in a finite space $\mathcal{S}$ called the state space, such that x_k depends only on s_k and y_k, and s_{k+1} is independent of $s_{<k}$ given s_k. Abusing notation the sequence $s_{1:k}$ is called the underlying Markov chain. A POMDP is called *ergodic* if there exists a policy such that the underlying Markov chain visits each state infinitely often with probability 1.

In particular, any ergodic POMDP ν satisfies strong α-mixing conditions with coefficients decaying exponentially fast in case there is a set $H\subset\mathcal{R}$ such that $\nu(r_i\in H)=1$ and $\nu(r_i=r|s_i=s,y_i=y)\neq 0$ for each $y\in\mathcal{Y},s\in\mathcal{S},r\in H,i\in I\!N$. Thus for any such POMDP ν we can use Proposition 7 with $d(k,\varepsilon)$ a constant function to show that ν is value-stable:

Corollary 8 (POMDP⇒value-stable)**.** *Suppose that a POMDP ν is ergodic and there exists a set $H \subset \mathcal{R}$ such that $\nu(r_i \in H) = 1$ and $\nu(r_i = r | s_i = s, y_i = y) \neq 0$ for each $y \in \mathcal{Y}, h \in \mathcal{S}, r \in H$, where $\mathcal{S}$ is the finite state space of the underlying Markov chain. Then ν is value-stable.*

However, it is illustrative to obtain this result for MDPs directly, and in a slightly stronger form.

Proposition 9 (MDP⇒value-stable)**.** *Any finite-state ergodic MDP ν is a value-stable environment.*

Proof. Let $d(k,\varepsilon) = 0$. Denote by μ the true environment, let $z_{<k}$ be the current history and let the current state (the observation x_k) of the environment be $a \in \mathcal{X}$, where $\mathcal{X}$ is the set of all possible states. Observe that for an MDP there is an optimal policy which depends only on the current state. Moreover, such a policy is optimal for any history. Let p_μ be such a policy. Let r_i^μ be the expected reward of p_μ on step i. Let $l(a,b) = \min\{n : x_{k+n} = b | x_k = a\}$. By ergodicity of μ there exists a policy p for which $\mathbf{E}l(b,a)$ is finite (and does not depend on k). A policy p needs to get from the state b to one of the states visited by an optimal policy, and then acts according to p_μ. Let $f(n) := \frac{nr_{\max}}{\log n}$. We have

$$
\begin{aligned}
&\mathbf{P}\left(\left|r^\mu_{k..k+n} - r^{p\mu}_{k..k+n}\right| > n\varepsilon\right) \le \sup_{a\in\mathcal{X}} \mathbf{P}\left(\left|\mathbf{E}\left(r^{p_\mu\mu}_{k..k+n}|x_k = a\right) - r^{p\mu}_{k..k+n}\right| > n\varepsilon\right)\right) \\
&\le \sup_{a,b\in\mathcal{X}} \mathbf{P}(l(a,b) > f(n)/r_{\max}) \\
&+ \sup_{a,b\in\mathcal{X}} \mathbf{P}\left(\left|\mathbf{E}\left(r^{p_\mu\mu}_{k..k+n}|x_k = a\right) - r^{p_\mu\mu}_{k+f(n)..k+n}\right| > n\varepsilon - f(n)\Big| x_{k+f(n)} = a\right) \\
&\le \sup_{a,b\in\mathcal{X}} \mathbf{P}(l(a,b) > f(n)/r_{\max}) \\
&+ \sup_{a\in\mathcal{X}} \mathbf{P}\left(\left|\mathbf{E}\left(r^{p_\mu\mu}_{k..k+n}|x_k = a\right) - r^{p_\mu\mu}_{k..k+n}\right| > n\varepsilon - 2f(n)\Big| x_k = a\right).
\end{aligned}
$$

In the last term we have the deviation of the reward attained by the optimal policy from its expectation. Clearly, both terms are bounded exponentially in n. □

In the examples above the function $d(k,\varepsilon)$ is a constant and $\varphi(n,\varepsilon)$ decays exponentially fast. This suggests that the class of value-stable environments stretches beyond finite (PO)MDPs. We illustrate this guess by the construction that follows.

A general scheme for constructing **value-stable environment or recoverable environments**: infinitely armed bandit. Next we present a construction of environments which cannot be modelled as finite POMDPs but are value-stable and/or recoverable. Consider the following environment ν. There is a countable family $\mathcal{C}' = \{\zeta_i : i \in I\!N\}$ of *arms*, that is, sources generating i.i.d. rewards 0 and 1 (and, say, empty observations) with some probability δ_i of the reward being 1. The action space $\mathcal{Y}$ consists of three actions $\mathcal{Y} = \{g,u,d\}$. To get the next reward from the current arm ζ_i an agent can use the action g. Let i denote the index of the current arm.

At the beginning $i=0$, the current arm is ζ_0 and then the agent can move between arms as follows: it can move $U(i)$ arms "up" using the action u (i.e. $i:=i+U(i)$) or it can move $D(i)$ arms "down" using the action d (i.e. $i:=i-D(i)$ or 0 if the result is negative). The reward for actions u and d is 0. In all the examples below $U(i)\equiv 1$, that is, the action u takes the agent one arm up.

Clearly, ν is a POMDP with countably infinite number of states in the underlying Markov chain, which (in general) is not isomorphic to a finite POMDP.

Claim 10. *If $D(i)=i$ for all $i\in I\!N$ then the environment ν just constructed is value-stable. If $D(i)\equiv 1$ then ν is recoverable but not necessarily value-stable; that is, there are choices of the probabilities δ_i such that ν is not value-stable.*

Proof. First we show that in either case ($D(i)=i$ or $D(i)=1$) ν is explorable. Let $\delta=\sup_{i\in I\!N}\delta_i$. Clearly, $\overline{V}(\nu,p')\leq\delta$ with probability 1 for any policy p' . A policy p which, knowing all the probabilities δ_i, achieves $\overline{V}(\nu,p)=\underline{V}(\nu,p)=\delta=:V^*_\nu$ a.s., can be easily constructed. Indeed, find a sequence ζ'_j, $j\in I\!N$, where for each j there is $i=:i_j$ such that $\zeta'_j=\zeta_i$, satisfying $\lim_{j\to\infty}\delta_{i_j}=\delta$. The policy p should carefully exploit one by one the arms ζ_j, staying with each arm long enough to ensure that the average reward is close to the expected reward with ε_j probability, where ε_j quickly tends to 0, and so that switching between arms has a negligible impact on the average reward. Thus ν can be shown to be explorable. Moreover, a policy p just sketched can be made independent on (observation and) rewards.

Next we show if $D(i)=i$, that is, the action d always takes the agent down to the first arm, then the environment is value-stable. Indeed, one can modify the policy p (possibly allowing it to exploit each arm longer) so that on each time step t (from some t on) we have $j(t)\leq\sqrt{t}$, where $j(t)$ is the number of the current arm on step t. Thus, after any actions-perceptions history $z_{<k}$ one needs about $\sqrt{k}$ actions (one action u and enough actions d) to catch up with p. So, (1) can be shown to hold with $d(k,\varepsilon)=\sqrt{k}$, r_i the expected reward of p on step i (since p is independent of rewards, $r_i^{p\nu}$ are independent), and the rates $\varphi(n,\varepsilon)$ exponential in n.

To construct a non-value-stable environment with $D(i)\equiv 1$, simply set $\delta_0=1$ and $\delta_j=0$ for $j>0$; then after taking n actions u one can only return to optimal rewards with n actions (d), that is $d(k)=o(k)$ cannot be obtained. Still it is easy to check that recoverability is preserved, whatever the choice of δ_i. □

In the above construction we can also allow the action d to bring the agent $d(i)<i$ steps down, where i is the number of the current environment ζ, according to some (possibly randomized) function $d(i)$, thus changing the function $d_\nu(k,\varepsilon)$ and possibly making it non-constant in ε and as close as desirable to linear.

7 Necessity of value-stability

Now we turn to the question of how tight the conditions of value-stability are. The following proposition shows that the requirement $d(k,\varepsilon)=o(k)$ in (1) cannot be relaxed.

Proposition 11 (necessity of $\boldsymbol{d(k,\varepsilon)=o(k)}$)**.** *There exists a countable family of deterministic explorable environments $\mathcal{C}$ such that*

- *for any $\nu\in\mathcal{C}$ for any sequence of actions $y_{<k}$ there exists a policy p such that*

$$r^{\nu}_{k..k+n} \le r^{p\nu}_{k..k+n} + k \text{ for all } n \ge k,$$

  where r^{ν}_i are the rewards attained by an optimal policy p_ν (which from the beginning was acting optimally), but

- *for any policy p there exists an environment $\nu\in\mathcal{C}$ such that $\underline{V}(\nu,p)<V^*_\nu$ (i.e. there is no self-optimizing policy for $\mathcal{C}$).*

Clearly, each environment from such a class $\mathcal{C}$ satisfies the value stability conditions with $\varphi(n,\varepsilon)\equiv 0$ except $d(k,\varepsilon)=k\neq o(k)$.

Proof. There are two possible actions $y_i\in\{a,b\}$, three possible rewards $r_i\in\{0,1,2\}$ and no observations.

Construct the environment ν_0 as follows: if $y_i=a$ then $r_i=1$ and if $y_i=b$ then $r_i=0$ for any $i\in I\!N$.

For each i let n_i denote the number of actions a taken up to step i: $n_i:=\#\{j\le i:y_j=a\}$. For each $s>0$ construct the environment ν_s as follows: $r_i(a)=1$ for any i, $r_i(b)=2$ if the longest consecutive sequence of action b taken has length greater than n_i and $n_i\ge s$; otherwise $r_i(b)=0$.

It is easy to see that each ν_i, $i>0$ satisfies the value stability conditions with $\varphi(n,\varepsilon)\equiv 0$ except $d(k,\varepsilon)=k\neq o(k)$, and does not satisfy it with any $d(k,\varepsilon)=o(k)$. Next we show that there is no self-optimizing policy for the class.

Suppose that there exists a policy p such that $\underline{V}(\nu_i,p)=V^*_{\nu_i}$ for each $i>0$ and let the true environment be ν_0. By assumption, for each s there exists such n that

$$\#\{i\le n: y_i=b, r_i=0\} \ge s > \#\{i\le n: y_i=a, r_i=1\}$$

which implies $\underline{V}(\nu_0,p)\le 1/2<1=V^*_{\nu_0}$. $\square$

It is also easy to show that the *uniformity of convergence in (1)* cannot be dropped. That is, if in the definition of value-stability we allow the function $\varphi(n,\varepsilon)$ to depend additionally on the past history $z_{<k}$ then Theorem 3 does not hold. This can be shown with the same example as constructed in the proof of Proposition 11, letting $d(k,\varepsilon)\equiv 0$ but instead allowing $\varphi(n,\varepsilon,z_{<k})$ to take values 0 and 1 according to the number of actions a taken, achieving the same behaviour as in the example provided in the last proof.

Moreover, we show that the requirement that the class $\mathcal{C}$ to be learnt is countable cannot be easily withdrawn. Indeed, consider the class of all deterministic passive environments in the sequence prediction setting. In this task an agent gets the reward 1 if $y_i = o_{i+1}$ and the reward 0 otherwise, where the sequence of observation o_i is deterministic. Different sequences correspond to different environments. As it was mentioned before, any such environment ν is value-stable with $d_\nu(k,\varepsilon) \equiv 1$, $\varphi_\nu(n,\varepsilon) \equiv 0$ and $r_i^\nu \equiv 1$. Obviously, the class of all deterministic passive environments is not countable. Since for every policy p there is an environment on which p errs exactly on each step, the class of all deterministic passive environments cannot be learned. Therefore, the following statement is valid:

Claim 12. *There exist (uncountable) classes of value-stable environments for which there are no self-optimizing policies.*

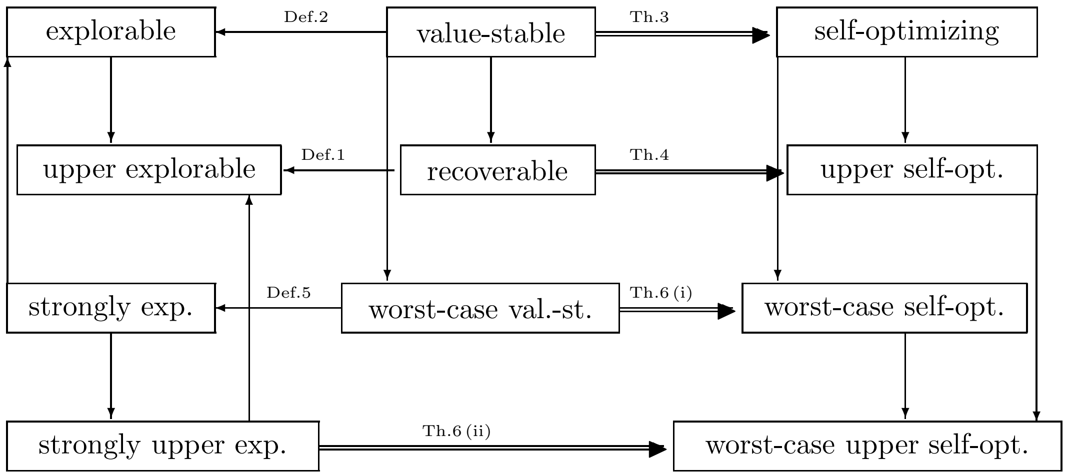
However, strictly speaking, even for countable classes value-stability is not necessary for self-optimizingness. This can be demonstrated on the class $\nu_i : i > 0$ from the proof of Proposition 11. (Whereas if we add ν_0 to the class a self-optimizing policy no longer exists.) So we have the following:

Claim 13. *There are countable classes of not value-stable environments for which self-optimizing policies exist.*

8 Discussion

Summary. We have proposed a set of conditions on environments, called value-stability, such that any countable class of value-stable environments admits a self-optimizing policy. It was also shown that these conditions are in a certain sense tight. The class of all value-stable environments includes ergodic MDPs, certain class of finite POMDPs, passive environments, and (provably) more environments. So the concept of value-stability allows us to characterize self-optimizing environment classes, and proving value-stability is typically much easier than proving self-optimizingness directly. Value stability means that from any (sup-optimal) sequence of actions it is possibly to recover fast. If it is possible to recover, but not necessarily fast, then we get a condition which we called recoverability, which was shown to be sufficient to be able to recover the upper limit of the optimal average asymptotic value. We have also analyzed what can be achieved in environments which possess (worst-case) value-stability but are not recoverable; it turned out that a certain worst-case self-optimizingness can be identified in this case too.

On the following picture we summarize the concepts introduced in Sections 3, 4 and 5. The arrows symbolize implications: some of them follow from theorems or stated in definitions (marked accordingly), while others are trivial.



Outlook. We considered only countable environment classes $\mathcal{C}$. From a computational perspective such classes are sufficiently large (e.g. the class of all computable probability measures is countable). On the other hand, countability excludes continuously parameterized families (like all ergodic MDPs), common in statistical practice. So perhaps the main open problem is to find under which conditions the requirement of countability of the class can be lifted. Another important question is whether (meaningful) necessary and sufficient conditions for self-optimizingness can be found. However, identifying classes of environments for which self-optimizing policies exist is a hard problem which has not been solved even for passive environments [RH06].

One more question concerns the uniformity of forgetfulness of the environment. Currently in the definition of value-stability (1) we have the function $\varphi(n,\varepsilon)$ which is the same for all histories $z_{<k}$, that is, both for all actions histories $y_{<k}$ and observations-rewards histories $x_{<k}$. Probably it is possible to differentiate between two types of forgetfulness, one for actions and one for perceptions.

In this work we have chosen the asymptotic uniform average value $\lim\frac{1}{m}r^{p\nu}_{1..m}$ as our performance measure. Another popular measure is the asymptotic discounted value $\gamma_1 r_1+\gamma_2 r_2+...$, where $\boldsymbol{\gamma}$ is some (typically geometric $\gamma_k\propto\gamma^k$) discount sequence. One can show [Hut06] under quite general conditions that the limit of average and future discounted values coincide. Equivalence holds for bounded rewards and monotone decreasing $\boldsymbol{\gamma}$, in deterministic environments and, in expectation over the history, also for probabilistic environments. So, in these cases our results also apply to discounted value.

Finally, it should be mentioned that we have concentrated on optimal values which can be obtained with certainty (with probability one); towards this aim we have defined (upper, strong) explorability and only considered environments which possess one of these properties. It would also be interesting to analyze what is achievable in environments which are not (upper, strongly) explorable; for example, one could consider optimal expected value, and may be some other criteria.

A Proofs of Theorems 3 and 4

In each of the proofs, a self-optimizing (or upper self-optimizing) policy p will be constructed. When the policy p has been defined up to a step k, an environment μ, endowed with this policy, can be considered as a measure on $\mathcal{Z}^k$. We assume this meaning when we use environments as measures on $\mathcal{Z}^k$ (e.g. $\mu(z_{<i})$).

Proof of Theorem 3. A self-optimizing policy p will be constructed as follows. On each step we will have two polices: p^t which exploits and p^e which explores; for each i the policy p either takes an action according to p^t ($p(z_{<i})=p^t(z_{<i})$) or according to p^e ($p(z_{<i})=p^e(z_{<i})$), as will be specified below.

In the algorithm below, i denotes the number of the current step in the sequence of actions-observations. Let $n=1$, $s=1$, and $j^t=j^e=0$. Let also $\alpha_s=2^{-s}$ for $s\in I\!N$. For each environment ν, find such a sequence of real numbers ε_n^ν that $\varepsilon_n^\nu\to 0$ and $\sum_{n=1}^\infty \varphi_\nu(n,\varepsilon_n^\nu)\leq\infty$.

Let $\imath: I\!N\to\mathcal{C}$ be such a numbering that each $\nu\in\mathcal{C}$ has infinitely many indices. For all $i>1$ define a measure ξ as follows

$$\xi(z_{<i}) = \sum_{\nu\in\mathcal{C}} w_\nu \nu(z_{<i}), \tag{3}$$

where $w_\nu\in\mathcal{R}$ are (any) such numbers that $\sum_\nu w_\nu=1$ and $w_\nu>0$ for all $\nu\in\mathcal{C}$.
Define $\boldsymbol{T}$. On each step i let

$$T \;\equiv\; T_i \;:=\; \left\{\nu\in\mathcal{C} : \frac{\nu(z_{<i})}{\xi(z_{<i})}\geq\alpha_s\right\}$$

Define $\boldsymbol{\nu^t}$. Set ν^t to be the first environment in T with index greater than $\imath(j^t)$. In case this is impossible (that is, if T is empty), increment s, (re)define T and try again. Increment j^t.
Define $\boldsymbol{\nu^e}$. Set ν^e to be the first environment with index greater than $\imath(j^e)$ such that $V^*_{\nu^e}>V^*_{\nu^t}$ and $\nu^e(z_{<k})>0$, if such an environment exists. Otherwise proceed one step (according to p^t) and try again. Increment j^e.
Consistency. On each step i (re)define T. If $\nu^t\notin T$, define ν^t, increment s and iterate the infinite loop. (Thus s is incremented only if ν^t is not in T or if T is empty.)

Start the **infinite loop**. Increment n.

Let $\delta:=(V^*_{\nu^e}-V^*_{\nu^t})/2$. Let $\varepsilon:=\varepsilon_n^{\nu^t}$. If $\varepsilon<\delta$ set $\delta=\varepsilon$. Let $h=j^e$.
Prepare for exploration.

Increment h. The index h is incremented with each next attempt of exploring ν^e. Each attempt will be at least h steps in length.

Let $p^t = p_{\nu^t}^{y_{<i}}$ and set $p = p^t$.

Let i_h be the current step. Find k_1 such that

$$\frac{i_h}{k_1} V^*_{\nu^t} \leq \varepsilon/8 \tag{4}$$

Find $k_2 > 2i_h$ such that for all $m > k_2$

$$\left| \frac{1}{m - i_h} r^{\nu^t}_{i_h+1..m} - V^*_{\nu^t} \right| \leq \varepsilon/8. \tag{5}$$

Find k_3 such that

$$hr_{max}/k_3 < \varepsilon/8. \tag{6}$$

Find k_4 such that for all $m > k_4$

$$\frac{1}{m} d_{\nu^e}(m, \varepsilon/4) \leq \varepsilon/8, \quad \frac{1}{m} d_{\nu^t}(m, \varepsilon/8) \leq \varepsilon/8 \text{ and } \frac{1}{m} d_{\nu^t}(i_h, \varepsilon/8) \leq \varepsilon/8. \tag{7}$$

Moreover, it is always possible to find such $k > \max\{k_1, k_2, k_3, k_4\}$ that

$$\frac{1}{2k} r^{\nu^e}_{k..3k} \geq \frac{1}{2k} r^{\nu^t}_{k..3k} + \delta. \tag{8}$$

Iterate up to the step k.

Exploration. Set $p^e = p^{y_{<n}}_{\nu^e}$. Iterate h steps according to $p = p^e$. Iterate further until either of the following conditions breaks

(i) $\left| r^{\nu^e}_{k..i} - r^{p\nu}_{k..i} \right| < (i-k)\varepsilon/4 + d_{\nu^e}(k, \varepsilon/4)$,

(ii) $i < 3k$.

(iii) $\nu^e \in T$.

Observe that either (i) or (ii) is necessarily broken.

If on some step ν^t is excluded from T then the infinite loop is iterated. If after exploration ν^e is not in T then redefine ν^e and **iterate the infinite loop**. If both ν^t and ν^e are still in T then **return** to "Prepare for exploration" (otherwise the loop is iterated with either ν^t or ν^e changed).

End of the infinite loop and the algorithm.

Let us show that with probability 1 the "Exploration" part is iterated only a finite number of times in a row with the same ν^t and ν^e.

Suppose the contrary, that is, suppose that (with some non-zero probability) the "Exploration" part is iterated infinitely often while $\nu^t, \nu^e \in T$. Observe that (1) implies that the ν^e-probability that (i) breaks is not greater than $\varphi_{\nu_e}(i-k, \varepsilon/4)$; hence by Borel-Cantelli lemma the event that (i) breaks infinitely often has probability 0 under ν^e.

Suppose that (i) holds almost every time. Then (ii) should be broken except for a finite number of times. We can use (4), (5), (7) and (8) to show that with probability at least $1-\varphi_{\nu^t}(k-i_h,\varepsilon/4)$ under ν^t we have $\frac{1}{3k}r^{p\nu^t}_{1..3k}\geq V^*_{\nu^t}+\varepsilon/2$. Again using Borel-Cantelli lemma and $k>2i_h$ we obtain that the event that (ii) breaks infinitely often has probability 0 under ν^t.

Thus (at least) one of the environments ν^t and ν^e is singular with respect to the true environment ν given the described policy and current history. Denote this environment by ν'. It is known (see e.g. [CS04, Thm.26]) that if measures μ and ν are mutually singular then $\frac{\mu(x_1,\dots,x_n)}{\nu(x_1,\dots,x_n)}\to\infty$ μ-a.s. Thus

$$\frac{\nu'(z_{<i})}{\nu(z_{<i})}\to 0 \ \nu\text{-a.s.} \tag{9}$$

Observe that (by definition of ξ) $\frac{\nu(z_{<i})}{\xi(z_{<i})}$ is bounded. Hence using (9) we can see that

$$\frac{\nu'(z_{<i})}{\xi(z_{<i})}\to 0 \ \nu\text{-a.s.}$$

Since s and α_s are not changed during the exploration phase this implies that on some step ν' will be excluded from T according to the "consistency" condition, which contradicts the assumption. Thus the "Exploration" part is iterated only a finite number of times in a row with the same ν^t and ν^e.

Observe that s is incremented only a finite number of times since $\frac{\nu'(z_{<i})}{\xi(z_{<i})}$ is bounded away from 0 where ν' is either the true environment ν or any environment from $\mathcal{C}$ which is equivalent to ν on the current history. The latter follows from the fact that $\frac{\xi(z_{<i})}{\nu(z_{<i})}$ is a submartingale with bounded expectation, and hence, by the submartingale convergence theorem (see e.g. [Doo53]) converges with ν-probability 1.

Let us show that from some step on ν (or an environment equivalent to it) is always in T and selected as ν^t. Consider the environment ν^t on some step i. If $V^*_{\nu^t}>V^*_\nu$ then ν^t will be excluded from T since on any optimal for ν^t sequence of actions (policy) measures ν and ν^t are singular. If $V^*_{\nu^t}<V^*_\nu$ than ν^e will be equal to ν at some point, and, after this happens sufficient number of times, ν^t will be excluded from T by the "exploration" part of the algorithm, s will be decremented and ν will be included into T. Finally, if $V^*_{\nu^t}=V^*_\nu$ then either the optimal value V^*_ν is (asymptotically) attained by the policy p_t of the algorithm, or (if p_{ν^t} is suboptimal for ν) $\frac{1}{i}r^{p\nu^t}_{1..i}<V^*_{\nu^t}-\varepsilon$ infinitely often for some ε, which has probability 0 under ν^t and consequently ν^t is excluded from T.

Thus, the exploration part ensures that all environments not equivalent to ν with indices smaller than $\imath(\nu)$ are removed from T and so from some step on ν^t is equal to (an environment equivalent to) the true environment ν.

We have shown in the "Exploration" part that $n\to\infty$, and so $\varepsilon^{\nu^t}_n\to 0$. Finally, using the same argument as before (Borel-Cantelli lemma, (i) and the definition of k) we can show that in the "exploration" and "prepare for exploration" parts of the algorithm the average value is within $\varepsilon^{\nu^t}_n$ of $V^*_{\nu^t}$ provided the true environment is (equivalent to) ν^t. □

Proof of Theorem 4. Let $\imath : I\!N \to \mathcal{C}$ be such a numbering that each $\nu \in \mathcal{C}$ has infinitely many indices. Define the measure ξ as in (3). The policy p acts according to the following algorithm.

Set $\varepsilon_s = \alpha_s = 2^{-s}$ for $s \in I\!N$, set $j = 1$, $s = n = 1$. The integer i will denote the current step in time.

Do the following *ad infinitum*. Set ν to be the first environment in $\mathcal{C}$ with index greater than $\imath(j)$. Find the policy p_ν which achieves the upper limiting average value with probability one (such policy exists by definition of recoverability). Act according to p_ν until either

$$\left| \frac{1}{i} r^{p\nu}_{1..i} - \overline{V}^*(p, p_\nu) \right| < \varepsilon_n \tag{10}$$

or

$$\frac{\nu(z_{<i})}{\xi(z_{<i})} < \alpha_s. \tag{11}$$

Increment n, s, i.

It can be easily seen that one of the conditions necessarily breaks. Indeed, either in the true environment the optimal upper limiting average value for the current environment ν can be achieved by the optimal policy p_ν, in which case (10) breaks; or it cannot be achieved, which means that ν and ξ are singular, which implies that (11) will be broken (see e.g. [CS04, Thm.26]; cf. the same argument in the proof of Theorem 3). Since ν equals the true environment infinitely often and $\varepsilon_n \to 0$ we get the statement of the theorem. □

Proof of Theorem 6 is analogous to the proofs of Theorems 3 and 4, except for the following. Instead of the optimal average value V^*_ν and upper optimal average value $\overline{V}^*_\nu$ the values $V^*_\nu(z_{<k})$ and $\overline{V}^*_\nu(z_{<k})$ should be used, and they should be updated after each step k. □

References

[Bos96] D. Bosq. *Nonparametric Statistics for Stochastic Processes.* Springer, 1996.

[BT99] R. I. Brafman and M. Tennenholtz. A general polynomial time algorithm for near-optimal reinforcement learning. In *Proc. 17th International Joint Conference on Artificial Intelligence (IJCAI-01)*, pages 734–739, 1999.

[CBL06] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games.* Cambridge University Press, 2006.

[CS04] I. Csiszar and P.C. Shields. Notes on information theory and statistics. In *Foundations and Trends in Communications and Information Theory*, 2004.

[dFM04] D. Pucci de Farias and N. Megiddo. How to combine expert (and novice) advice when actions impact the environment? In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16.* MIT Press, Cambridge, MA, 2004.

[Doo53] J. L. Doob. *Stochastic Processes.* John Wiley & Sons, New York, 1953.

[EDKM05] E. Even-Dar, S. M. Kakade, and Y. Mansour. Reinforcement learning in POMDPs without resets. In *IJCAI*, pages 690–695, 2005.

[HP04] M. Hutter and J. Poland. Prediction with expert advice by following the perturbed leader for general weights. In *Proc. 15th International Conf. on Algorithmic Learning Theory (ALT'04)*, volume 3244 of *LNAI*, pages 279–293, Padova, 2004. Springer, Berlin.

[Hut02] M. Hutter. Self-optimizing and Pareto-optimal policies in general environments based on Bayes-mixtures. In *Proc. 15th Annual Conference on Computational Learning Theory (COLT 2002)*, Lecture Notes in Artificial Intelligence, pages 364–379, Sydney, Australia, July 2002. Springer.

[Hut03] M. Hutter. Optimality of universal Bayesian prediction for general loss and alphabet. *Journal of Machine Learning Research*, 4:971–1000, 2003.

[Hut05] M. Hutter. *Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability.* Springer, Berlin, 2005. 300 pages, http://www.hutter1.net/ai/uaibook.htm.

[Hut06] M. Hutter. General discounting versus average reward. In *Proc. 17th International Conf. on Algorithmic Learning Theory (ALT'06)*, volume 4264 of *LNAI*, pages 244–258, Barcelona, 2006. Springer, Berlin.

[KV86] P. R. Kumar and P. P. Varaiya. *Stochastic Systems: Estimation, Identification, and Adaptive Control.* Prentice Hall, Englewood Cliffs, NJ, 1986.

[PH05] J. Poland and M. Hutter. Defensive universal learning with experts. In *Proc. 16th International Conf. on Algorithmic Learning Theory (ALT'05)*, volume 3734 of *LNAI*, pages 356–370, Singapore, 2005. Springer, Berlin.

[PH06] J. Poland and M. Hutter. Universal learning of repeated matrix games. In *Annual Machine Learning Conference of Belgium and The Netherlands (Benelearn'06)*, pages 7–14, Ghent, 2006.

[RH06] D. Ryabko and M. Hutter. Predicting Non-Stationary Processes. Applied Mathematics Letters, 21:5, pages 477–482, 2008.

[RN95] S. J. Russell and P. Norvig. *Artificial Intelligence. A Modern Approach.* Prentice-Hall, Englewood Cliffs, 1995.

[SB98] R. Sutton and A. Barto. *Reinforcement learning: An introduction.* Cambridge, MA, MIT Press, 1998.