A Formal Solution to the Grain of Truth Problem

Jan Leike
Australian National University
jan.leike@anu.edu.au

Jessica Taylor
Machine Intelligence Research Inst.
jessica@intelligence.org

Benya Fallenstein
Machine Intelligence Research Inst.
benya@intelligence.org

Abstract

A Bayesian agent acting in a multi-agent environment learns to predict the other agents' policies if its prior assigns positive probability to them (in other words, its prior contains a *grain of truth*). Finding a reasonably large class of policies that contains the Bayes-optimal policies with respect to this class is known as the *grain of truth problem*. Only small classes are known to have a grain of truth and the literature contains several related impossibility results. In this paper we present a formal and general solution to the full grain of truth problem: we construct a class of policies that contains all computable policies as well as Bayes-optimal policies for every lower semicomputable prior over the class. When the environment is unknown, Bayes-optimal agents may fail to act optimally even asymptotically. However, agents based on Thompson sampling converge to play ε-Nash equilibria in arbitrary unknown computable multi-agent environments. While these results are purely theoretical, we show that they can be computationally approximated arbitrarily closely.

Keywords. General reinforcement learning, multi-agent systems, game theory, self-reflection, asymptotic optimality, Nash equilibrium, Thompson sampling, AIXI.

1 INTRODUCTION

Consider the general setup of multiple reinforcement learning agents interacting sequentially in a known environment with the goal to maximize discounted reward.[1] Each agent knows how the environment behaves, but does not know the other agents' behavior. The natural (Bayesian) approach would be to define a class of possible policies that the other agents could adopt and take a prior over this class. During the interaction, this prior gets updated to the posterior as our agent learns the others' behavior. Our agent then acts optimally with respect to this posterior belief.

Reinforcement learning	Game theory
stochastic policy	mixed strategy
deterministic policy	pure strategy
agent	player
multi-agent environment	infinite extensive-form game
reward	payoff/utility
(finite) history	history
infinite history	path of play

Table 1: Terminology dictionary between reinforcement learning and game theory.

A famous result for infinitely repeated games states that as long as each agent assigns positive prior probability to the other agents' policies (a *grain of truth*) and each agent acts Bayes-optimal, then the agents converge to playing an ε-Nash equilibrium [KL93].

As an example, consider an infinitely repeated prisoners dilemma between two agents. In every time step the payoff matrix is as follows, where C means cooperate and D means defect.

	C	D
C	3/4, 3/4	0, 1
D	1, 0	1/4, 1/4

Define the set of policies $\Pi := \{\pi_\infty, \pi_0, \pi_1, \ldots\}$ where policy π_t cooperates until time step t or the opponent defects (whatever happens first) and defects thereafter. The Bayes-optimal behavior is to cooperate until the posterior belief that the other agent defects in the time step after the next is greater than some constant (depending on the discount function) and then defect afterwards. Therefore Bayes-optimal behavior leads to a policy from the set Π

[1] We mostly use the terminology of reinforcement learning. For readers from game theory we provide a dictionary in Table 1.

(regardless of the prior). If both agents are Bayes-optimal with respect to some prior, they both have a grain of truth and therefore they converge to a Nash equilibrium: either they both cooperate forever or after some finite time they both defect forever. Alternating strategies like TitForTat (cooperate first, then play the opponent's last action) are not part of the policy class Π, and adding them to the class breaks the grain of truth property: the Bayes-optimal behavior is no longer in the class. This is rather typical; a Bayesian agent usually needs to be more powerful than its environment [LH15b].

Until now, classes that admit a grain of truth were known only for small toy examples such as the iterated prisoner's dilemma above [SLB09, Ch. 7.3]. The quest to find a large class admitting a grain of truth is known as the *grain of truth problem* [Hut09, Q. 5j]. The literature contains several impossibility results on the grain of truth problem [FY01, Nac97, Nac05] that identify properties that cannot be simultaneously satisfied for classes that allow a grain of truth.

In this paper we present a formal solution to multi-agent reinforcement learning and the grain of truth problem in the general setting (Section 3). We assume that our multi-agent environment is computable, but it does not need to be stationary/Markov, ergodic, or finite-state [Hut05]. Our class of policies is large enough to contain all computable (stochastic) policies, as well as all relevant Bayes-optimal policies. At the same time, our class is small enough to be limit computable. This is important because it allows our result to be computationally approximated.

In Section 4 we consider the setting where the multi-agent environment is unknown to the agents and has to be learned in addition to the other agents' behavior. A Bayes-optimal agent may not learn to act optimally in unknown multi-agent environments *even though it has a grain of truth*. This effect occurs in non-recoverable environments where taking one wrong action can mean a permanent loss of future value. In this case, a Bayes-optimal agent avoids taking these dangerous actions and therefore will not explore enough to wash out the prior's bias [LH15a]. Therefore, Bayesian agents are not *asymptotically optimal*, i.e., they do not always learn to act optimally [Ors13].

However, asymptotic optimality is achieved by Thompson sampling because the inherent randomness of Thompson sampling leads to enough exploration to learn the entire environment class [LLOH16]. This leads to our main result: if all agents use Thompson sampling over our class of multi-agent environments, then for every $\varepsilon > 0$ they converge to an ε-Nash equilibrium asymptotically.

The central idea to our construction is based on *reflective oracles* [FST15, FTC15b]. Reflective oracles are probabilistic oracles similar to halting oracles that answer whether the probability that a given probabilistic Turing machine T outputs 1 is higher than a given rational number p. The oracles are reflective in the sense that the machine T may itself query the oracle, so the oracle has to answer queries about itself. This invites issues caused by self-referential liar paradoxes of the form "if the oracle says that I return 1 with probability $> 1/2$, then return 0, else return 1." Reflective oracles avoid these issues by being allowed to randomize if the machines do not halt or the rational number is *exactly* the probability to output 1. We introduce reflective oracles formally in Section 2 and prove that there is a limit computable reflective oracle.

2 REFLECTIVE ORACLES

2.1 PRELIMINARIES

Let $\mathcal{X}$ denote a finite set called *alphabet*. The set $\mathcal{X}^* := \bigcup_{n=0}^{\infty} \mathcal{X}^n$ is the set of all finite strings over the alphabet $\mathcal{X}$, the set $\mathcal{X}^\infty$ is the set of all infinite strings over the alphabet $\mathcal{X}$, and the set $\mathcal{X}^\sharp := \mathcal{X}^* \cup \mathcal{X}^\infty$ is their union. The empty string is denoted by ϵ, not to be confused with the small positive real number ε. Given a string $x \in \mathcal{X}^\sharp$, we denote its length by $|x|$. For a (finite or infinite) string x of length $\geq k$, we denote with $x_{1:k}$ the first k characters of x, and with $x_{<k}$ the first $k-1$ characters of x. The notation $x_{1:\infty}$ stresses that x is an infinite string.

A function $f : \mathcal{X}^* \to \mathbb{R}$ is *lower semicomputable* iff the set $\{(x,p) \in \mathcal{X}^* \times \mathbb{Q} \mid f(x) > p\}$ is recursively enumerable. The function f is *computable* iff both f and $-f$ are lower semicomputable. Finally, the function f is *limit computable* iff there is a computable function ϕ such that

$$\lim_{k\to\infty} \phi(x,k) = f(x).$$

The program ϕ that limit computes f can be thought of as an *anytime algorithm* for f: we can stop ϕ at any time k and get a preliminary answer. If the program ϕ ran long enough (which we do not know), this preliminary answer will be close to the correct one.

We use $\Delta\mathcal{Y}$ to denote the set of probability distributions over $\mathcal{Y}$. A list of notation can be found in Appendix A.

2.2 DEFINITION

A *semimeasure* over the alphabet $\mathcal{X}$ is a function $\nu : \mathcal{X}^* \to [0,1]$ such that (i) $\nu(\epsilon) \leq 1$, and (ii) $\nu(x) \geq \sum_{a\in\mathcal{X}} \nu(xa)$ for all $x \in \mathcal{X}^*$. In the terminology of measure theory, semimeasures are probability measures on the probability space $\mathcal{X}^\sharp = \mathcal{X}^* \cup X^\infty$ whose σ-algebra is generated by the *cylinder sets* $\Gamma_x := \{xz \mid z \in \mathcal{X}^\sharp\}$ [LV08, Ch. 4.2]. We call a semimeasure (probability) a *measure* iff equalities hold in (i) and (ii) for all $x \in \mathcal{X}^*$.

Next, we connect semimeasures to Turing machines. The literature uses *monotone Turing machines*, which naturally

correspond to lower semicomputable semimeasures [LV08, Sec. 4.5.2] that describe the distribution that arises when piping fair coin flips into the monotone machine. Here we take a different route.

A *probabilistic Turing machine* is a Turing machine that has access to an unlimited number of uniformly random coin flips. Let $\mathcal{T}$ denote the set of all probabilistic Turing machines that take some input in $\mathcal{X}^*$ and may query an oracle (formally defined below). We take a Turing machine $T \in \mathcal{T}$ to correspond to a semimeasure λ_T where $\lambda_T(a \mid x)$ is the probability that T outputs $a \in \mathcal{X}$ when given $x \in \mathcal{X}^*$ as input. The value of $\lambda_T(x)$ is then given by the chain rule

$$\lambda_T(x) := \prod_{k=1}^{|x|} \lambda_T(x_k \mid x_{<k}). \tag{1}$$

Thus $\mathcal{T}$ gives rise to the set of semimeasures $\mathcal{M}$ where the *conditionals* $\lambda(a \mid x)$ are lower semicomputable. In contrast, the literature typically considers semimeasures whose *joint* probability (1) is lower semicomputable. This set $\mathcal{M}$ contains all computable measures. However, $\mathcal{M}$ is a proper subset of the set of all lower semicomputable semimeasures because the product (1) is lower semicomputale, but there are some lower semicomputable semimeasures whose conditional is not lower semicomputable [LH15c, Thm. 6].

In the following we assume that our alphabet is binary, i.e., $\mathcal{X} := \{0,1\}$.

Definition 1 (Oracle)**.** An *oracle* is a function $O : \mathcal{T} \times \{0,1\}^* \times \mathbb{Q} \to \Delta\{0,1\}$.

Oracles are understood to be probabilistic: they randomly return 0 or 1. Let T^O denote the machine $T \in \mathcal{T}$ when run with the oracle O, and let λ_T^O denote the semimeasure induced by T^O. This means that drawing from λ_T^O involves two sources of randomness: one from the distribution induced by the probabilistic Turing machine T and one from the oracle's answers.

The intended semantics of an oracle are that it takes a *query* (T, x, p) and returns 1 if the machine T^O outputs 1 on input x with probability greater than p when run with the oracle O, i.e., when $\lambda_T^O(1 \mid x) > p$. Furthermore, the oracle returns 0 if the machine T^O outputs 1 on input x with probability less than p when run with the oracle O, i.e., when $\lambda_T^O(1 \mid x) < p$. To fulfill this, the oracle O has to make statements about itself, since the machine T from the query may again query O. Therefore we call oracles of this kind *reflective oracles*. This has to be defined very carefully to avoid the obvious diagonalization issues that are caused by programs that ask the oracle about themselves. We impose the following self-consistency constraint.

Definition 2 (Reflective Oracle)**.** An oracle O is *reflective* iff for all queries $(T, x, p) \in \mathcal{T} \times \{0,1\}^* \times \mathbb{Q}$,

(i) $\lambda_T^O(1 \mid x) > p$ implies $O(T, x, p) = 1$, and

Figure 1: Answer options of a reflective oracle O for the query (T, x, p); the rational $p \in [0,1]$ falls into one of the three regions above. The values of $\lambda_T^O(0 \mid x)$ and $\lambda_T^O(1 \mid x)$ are depicted as the length of the line segment under which they are written.

(ii) $\lambda_T^O(0 \mid x) > 1 - p$ implies $O(T, x, p) = 0$.

If p under- or overshoots the true probability of $\lambda_T^O(\cdot \mid x)$, then the oracle must reveal this information. However, in the critical case when $p = \lambda_T^O(1 \mid x)$, the oracle is allowed to return anything and may randomize its result. Furthermore, since T might not output any symbol, it is possible that $\lambda_T^O(0 \mid x) + \lambda_T^O(1 \mid x) < 1$. In this case the oracle can reassign the non-halting probability mass to 0, 1, or randomize; see Figure 1.

Example 3 (Reflective Oracles and Diagonalization)**.** Let $T \in \mathcal{T}$ be a probabilistic Turing machine that outputs $1 - O(T, \epsilon, 1/2)$ (T can know its own source code by quining [Kle52, Thm. 27]). In other words, T queries the oracle about whether it is more likely to output 1 or 0, and then does whichever the oracle says is less likely. In this case we can use an oracle $O(T, \epsilon, 1/2) := 1/2$ (answer 0 or 1 with equal probability), which implies $\lambda_T^O(1 \mid \epsilon) = \lambda_T^O(0 \mid \epsilon) = 1/2$, so the conditions of Definition 2 are satisfied. In fact, for this machine T we must have $O(T, \epsilon, 1/2) = 1/2$ for all reflective oracles O. ♢

The following theorem establishes that reflective oracles exist.

Theorem 4 ([FTC15a, App. B])**.** *There is a reflective oracle.*

Definition 5 (Reflective-Oracle-Computable)**.** A semimeasure is called *reflective-oracle-computable* iff it is computable on a probabilistic Turing machine with access to a reflective oracle.

For any probabilistic Turing machine $T \in \mathcal{T}$ we can complete the semimeasure $\lambda_T^O(\cdot \mid x)$ into a reflective-oracle-computable measure $\overline{\lambda}_T^O(\cdot \mid x)$: Using the oracle O and a binary search on the parameter p we search for the crossover point p where $O(T, x, p)$ goes from returning 1 to returning 0. The limit point $p^* \in \mathbb{R}$ of the binary search is random since the oracle's answers may be random. But the main point is that the expectation of p^* exists, so $\overline{\lambda}_T^O(1 \mid x) = \mathbb{E}[p^*] = 1 - \overline{\lambda}_T^O(0 \mid x)$ for all $x \in \mathcal{X}^*$. Hence $\overline{\lambda}_T^O$ is a measure. Moreover, if the oracle is reflective, then $\overline{\lambda}_T^O(x) \geq \lambda_T^O(x)$ for all $x \in \mathcal{X}^*$. In this sense the oracle O can be viewed as a way of 'completing'

all semimeasures λ_T^O to measures by arbitrarily assigning the non-halting probability mass. If the oracle O is reflective this is consistent in the sense that Turing machines who run other Turing machines will be completed in the same way. This is especially important for a universal machine that runs all other Turing machines to induce a Solomonoff-style distribution.

2.3 A LIMIT COMPUTABLE REFLECTIVE ORACLE

The proof of Theorem 4 given in [FTC15a, App. B] is non-constructive and uses the axiom of choice. In Section 2.4 we give a constructive proof for the existence of reflective oracles and show that there is one that is limit computable.

Theorem 6 (A Limit Computable Reflective Oracle)**.** *There is a reflective oracle that is limit computable.*

This theorem has the immediate consequence that reflective oracles cannot be used as halting oracles. At first, this result may seem surprising: according to the definition of reflective oracles, they make concrete statements about the output of probabilistic Turing machines. However, the fact that the oracles may randomize some of the time actually removes enough information such that halting can no longer be decided from the oracle output.

Corollary 7 (Reflective Oracles are not Halting Oracles)**.** *There is no probabilistic Turing machine T such that for every prefix program p and every reflective oracle O, we have that $\lambda_T^O(1 \mid p) > 1/2$ if p halts and $\lambda_T^O(1 \mid p) < 1/2$ otherwise.*

Proof. Assume there was such a machine T and let O be the limit computable oracle from Theorem 6. Since O is reflective we can turn T into a deterministic halting oracle by calling $O(T, p, 1/2)$ which deterministically returns 1 if p halts and 0 otherwise. Since O is limit computable, we can finitely compute the output of O on any query to arbitrary finite precision using our deterministic halting oracle. We construct a probabilistic Turing machine T' that uses our halting oracle to compute (rather than query) the oracle O on $(T', \epsilon, 1/2)$ to a precision of $1/3$ in finite time. If $O(T', \epsilon, 1/2) \pm 1/3 > 1/2$, the machine T' outputs 0, otherwise T' outputs 1. Since our halting oracle is entirely deterministic, the output of T' is entirely deterministic as well (and T' always halts), so $\lambda_{T'}^O(0 \mid \epsilon) = 1$ or $\lambda_{T'}^O(1 \mid \epsilon) = 1$. Therefore $O(T', \epsilon, 1/2) = 1$ or $O(T', \epsilon, 1/2) = 0$ because O is reflective. A precision of $1/3$ is enough to tell them apart, hence T' returns 0 if $O(T', \epsilon, 1/2) = 1$ and T' returns 1 if $O(T', \epsilon, 1/2) = 0$. This is a contradiction. □

A similar argument can also be used to show that reflective oracles are not computable.

2.4 PROOF OF THEOREM 6

The idea for the proof of Theorem 6 is to construct an algorithm that outputs an infinite series of *partial oracles* converging to a reflective oracle in the limit.

The set of queries is countable, so we can assume that we have some computable enumeration of it:

$$\mathcal{T} \times \{0,1\}^* \times \mathbb{Q} =: \{q_1, q_2, \ldots\}$$

Definition 8 (k-Partial Oracle)**.** A *k-partial oracle* $\tilde{O}$ is function from the first k queries to the multiples of 2^{-k} in $[0, 1]$:

$$\tilde{O} : \{q_1, q_2, \ldots, q_k\} \to \{n2^{-k} \mid 0 \leq n \leq 2^k\}$$

Definition 9 (Approximating an Oracle)**.** A k-partial oracle $\tilde{O}$ *approximates* an oracle O iff $|O(q_i) - \tilde{O}(q_i)| \leq 2^{-k-1}$ for all $i \leq k$.

Let $k \in \mathbb{N}$, let $\tilde{O}$ be a k-partial oracle, and let $T \in \mathcal{T}$ be an oracle machine. The machine $T^{\tilde{O}}$ that we get when we run T with the k-partial oracle $\tilde{O}$ is defined as follows (this is with slight abuse of notation since k is taken to be understood implicitly).

1. Run T for at most k steps.
2. If T calls the oracle on q_i for $i \leq k$,
   (a) return 1 with probability $\tilde{O}(q_i) - 2^{-k-1}$,
   (b) return 0 with probability $1 - \tilde{O}(q_i) - 2^{-k-1}$, and
   (c) halt otherwise.
3. If T calls the oracle on q_j for $j > k$, halt.

Furthermore, we define $\lambda_T^{\tilde{O}}$ analogously to λ_T^O as the distribution generated by the machine $T^{\tilde{O}}$.

Lemma 10. *If a k-partial oracle $\tilde{O}$ approximates a reflective oracle O, then $\lambda_T^O(1 \mid x) \geq \lambda_T^{\tilde{O}}(1 \mid x)$ and $\lambda_T^O(0 \mid x) \geq \lambda_T^{\tilde{O}}(0 \mid x)$ for all $x \in \{0,1\}^*$ and all $T \in \mathcal{T}$.*

Proof. This follows from the definition of $T^{\tilde{O}}$: when running T with $\tilde{O}$ instead of O, we can only lose probability mass. If T makes calls whose index is $> k$ or runs for more than k steps, then the execution is aborted and no further output is generated. If T makes calls whose index $i \leq k$, then $\tilde{O}(q_i) - 2^{-k-1} \leq O(q_i)$ since $\tilde{O}$ approximates O. Therefore the return of the call q_i is underestimated as well. □

Definition 11 (k-Partially Reflective)**.** A k-partial oracle $\tilde{O}$ is *k-partially reflective* iff for the first k queries (T, x, p)

- $\lambda_T^{\tilde{O}}(1 \mid x) > p$ implies $\tilde{O}(T, x, p) = 1$, and

- $\lambda_T^{\tilde{O}}(0 \mid x) > 1 - p$ implies $\tilde{O}(T, x, p) = 0$.

It is important to note that we can check whether a k-partial oracle is k-partially reflective in finite time by running all machines T from the first k queries for k steps and tallying up the probabilities to compute $\lambda_T^{\tilde{O}}$.

Lemma 12. *If O is a reflective oracle and $\tilde{O}$ is a k-partial oracle that approximates O, then $\tilde{O}$ is k-partially reflective.*

Lemma 12 only holds because we use semimeasures whose conditionals are lower semicomputable.

Proof. Assuming $\lambda_T^{\tilde{O}}(1 \mid x) > p$ we get from Lemma 10 that $\lambda_T^{O}(1 \mid x) \geq \lambda_T^{\tilde{O}}(1 \mid x) > p$. Thus $O(T, x, p) = 1$ because O is reflective. Since $\tilde{O}$ approximates O, we get $1 = O(T, x, p) \leq \tilde{O}(T, x, p) + 2^{-k-1}$, and since $\tilde{O}$ assigns values in a 2^{-k}-grid, it follows that $\tilde{O}(T, x, p) = 1$. The second implication is proved analogously. □

Definition 13 (Extending Partial Oracles)**.** A $k+1$-partial oracle $\tilde{O}'$ *extends* a k-partial oracle $\tilde{O}$ iff $|\tilde{O}(q_i) - \tilde{O}'(q_i)| \leq 2^{-k-1}$ for all $i \leq k$.

Lemma 14. *There is an infinite sequence of partial oracles $(\tilde{O}_k)_{k \in \mathbb{N}}$ such that for each k, $\tilde{O}_k$ is a k-partially reflective k-partial oracle and $\tilde{O}_{k+1}$ extends $\tilde{O}_k$.*

Proof. By Theorem 4 there is a reflective oracle O. For every k, there is a canonical k-partial oracle $\tilde{O}_k$ that approximates O: restrict O to the first k queries and for any such query q pick the value in the 2^{-k}-grid which is closest to $O(q)$. By construction, $\tilde{O}_{k+1}$ extends $\tilde{O}_k$ and by Lemma 12, each $\tilde{O}_k$ is k-partially reflective. □

Lemma 15. *If the $k+1$-partial oracle $\tilde{O}_{k+1}$ extends the k-partial oracle $\tilde{O}_k$, then $\lambda_T^{\tilde{O}_{k+1}}(1 \mid x) \geq \lambda_T^{\tilde{O}_k}(1 \mid x)$ and $\lambda_T^{\tilde{O}_{k+1}}(0 \mid x) \geq \lambda_T^{\tilde{O}_k}(0 \mid x)$ for all $x \in \{0,1\}^*$ and all $T \in \mathcal{T}$.*

Proof. $T^{\tilde{O}_{k+1}}$ runs for one more step than $T^{\tilde{O}_k}$, can answer one more query and has increased oracle precision. Moreover, since $\tilde{O}_{k+1}$ extends $\tilde{O}_k$, we have $|\tilde{O}_{k+1}(q_i) - \tilde{O}_k(q_i)| \leq 2^{-k-1}$, and thus $\tilde{O}_{k+1}(q_i) - 2^{-k-1} \geq \tilde{O}_k(q_i) - 2^{-k}$. Therefore the success to answers to the oracle calls (case 2(a) and 2(b)) will not decrease in probability. □

Now everything is in place to state the algorithm that constructs a reflective oracle in the limit. It recursively traverses a tree of partial oracles. The tree's nodes are the partial oracles; level k of the tree contains all k-partial oracles. There is an edge in the tree from the k-partial oracle $\tilde{O}_k$ to the i-partial oracle $\tilde{O}_i$ if and only if $i = k+1$ and $\tilde{O}_i$ extends $\tilde{O}_k$.

For every k, there are only finitely many k-partial oracles, since they are functions from finite sets to finite sets. In particular, there are exactly two 1-partial oracles (so the search tree has two roots). Pick one of them to start with, and proceed recursively as follows. Given a k-partial oracle $\tilde{O}_k$, there are finitely many $(k+1)$-partial oracles that extend $\tilde{O}_k$ (finite branching of the tree). Pick one that is $(k+1)$-partially reflective (which can be checked in finite time). If there is no $(k+1)$-partially reflective extension, backtrack.

By Lemma 14 our search tree is infinitely deep and thus the tree search does not terminate. Moreover, it can backtrack to each level only a finite number of times because at each level there is only a finite number of possible extensions. Therefore the algorithm will produce an infinite sequence of partial oracles, each extending the previous. Because of finite backtracking, the output eventually stabilizes on a sequence of partial oracles $\tilde{O}_1, \tilde{O}_2, \ldots$. By the following lemma, this sequence converges to a reflective oracle, which concludes the proof of Theorem 6.

Lemma 16. *Let $\tilde{O}_1, \tilde{O}_2, \ldots$ be a sequence where $\tilde{O}_k$ is a k-partially reflective k-partial oracle and $\tilde{O}_{k+1}$ extends $\tilde{O}_k$ for all $k \in \mathbb{N}$. Let $O := \lim_{k \to \infty} \tilde{O}_k$ be the pointwise limit. Then*

(a) $\lambda_T^{\tilde{O}_k}(1 \mid x) \to \lambda_T^{O}(1 \mid x)$ and $\lambda_T^{\tilde{O}_k}(0 \mid x) \to \lambda_T^{O}(0 \mid x)$ as $k \to \infty$ for all $x \in \{0,1\}^*$ and all $T \in \mathcal{T}$, and

(b) O is a reflective oracle.

Proof. First note that the pointwise limit must exists because $|\tilde{O}_k(q_i) - \tilde{O}_{k+1}(q_i)| \leq 2^{-k-1}$ by Definition 13.

(a) Since $\tilde{O}_{k+1}$ extends $\tilde{O}_k$, each $\tilde{O}_k$ approximates O. Let $x \in \{0,1\}^*$ and $T \in \mathcal{T}$ and consider the sequence $a_k := \lambda_T^{\tilde{O}_k}(1 \mid x)$ for $k \in \mathbb{N}$. By Lemma 15, $a_k \leq a_{k+1}$, so the sequence is monotone increasing. By Lemma 10, $a_k \leq \lambda_T^{O}(1 \mid x)$, so the sequence is bounded. Therefore it must converge. But it cannot converge to anything strictly below $\lambda_T^{O}(1 \mid x)$ by the definition of T^O.

(b) By definition, O is an oracle; it remains to show that O is reflective. Let $q_i = (T, x, p)$ be some query. If $p < \lambda_T^{O}(1 \mid x)$, then by (a) there is a k large enough such that $p < \lambda_T^{\tilde{O}_t}(1 \mid x)$ for all $t \geq k$. For any $t \geq \max\{k, i\}$, we have $\tilde{O}_t(T, x, p) = 1$ since $\tilde{O}_t$ is t-partially reflective. Therefore $1 = \lim_{k \to \infty} \tilde{O}_k(T, x, p) = O(T, x, p)$. The case $1 - p < \lambda_T^{O}(0 \mid x)$ is analogous. □

3 A GRAIN OF TRUTH

3.1 NOTATION

In reinforcement learning, an agent interacts with an environment in cycles: at time step t the agent chooses an *action* $a_t \in \mathcal{A}$ and receives a *percept* $e_t = (o_t, r_t) \in \mathcal{E}$

consisting of an *observation* $o_t \in \mathcal{O}$ and a real-valued *reward* $r_t \in \mathbb{R}$; the cycle then repeats for $t+1$. A *history* is an element of $(\mathcal{A} \times \mathcal{E})^*$. In this section, we use $æ \in \mathcal{A} \times \mathcal{E}$ to denote one interaction cycle, and $æ_{<t}$ to denote a history of length $t-1$.

We fix a *discount function* $\gamma : \mathbb{N} \to \mathbb{R}$ with $\gamma_t \geq 0$ and $\sum_{t=1}^{\infty} \gamma_t < \infty$. The goal in reinforcement learning is to maximize discounted rewards $\sum_{t=1}^{\infty} \gamma_t r_t$. The *discount normalization factor* is defined as $\Gamma_t := \sum_{k=t}^{\infty} \gamma_k$. The *effective horizon* $H_t(\varepsilon)$ is a horizon that is long enough to encompass all but an ε of the discount function's mass:

$$H_t(\varepsilon) := \min\{k \mid \Gamma_{t+k}/\Gamma_t \leq \varepsilon\} \tag{2}$$

A *policy* is a function $\pi : (\mathcal{A} \times \mathcal{E})^* \to \Delta\mathcal{A}$ that maps a history $æ_{<t}$ to a distribution over actions taken after seeing this history. The probability of taking action a after history $æ_{<t}$ is denoted with $\pi(a \mid æ_{<t})$. An *environment* is a function $\nu : (\mathcal{A} \times \mathcal{E})^* \times \mathcal{A} \to \Delta\mathcal{E}$ where $\nu(e \mid æ_{<t}a_t)$ denotes the probability of receiving the percept e when taking the action a_t after the history $æ_{<t}$. Together, a policy π and an environment ν give rise to a distribution ν^π over histories. Throughout this paper, we make the following assumptions.

Assumption 17. *(a) Rewards are bounded between* 0 *and* 1.

(b) The set of actions $\mathcal{A}$ and the set of percepts $\mathcal{E}$ are both finite.

(c) The discount function γ and the discount normalization factor Γ are computable.

Definition 18 (Value Function)**.** The *value* of a policy π in an environment ν given history $æ_{<t}$ is defined recursively as $V_\nu^\pi(æ_{<t}) := \sum_{a \in \mathcal{A}} \pi(a \mid æ_{<t}) V_\nu^\pi(æ_{<t}a)$ and

$$V_\nu^\pi(æ_{<t}a_t) := \frac{1}{\Gamma_t} \sum_{e_t \in \mathcal{E}} \nu(e_t \mid æ_{<t}a_t)\Big(\gamma_t r_t + \Gamma_{t+1} V_\nu^\pi(æ_{1:t})\Big)$$

if $\Gamma_t > 0$ and $V_\nu^\pi(æ_{<t}a_t) := 0$ if $\Gamma_t = 0$. The *optimal value* is defined as $V_\nu^*(æ_{<t}) := \sup_\pi V_\nu^\pi(æ_{<t})$.

Definition 19 (Optimal Policy)**.** A policy π is *optimal in environment ν (ν-optimal)* iff for all histories $æ_{<t} \in (\mathcal{A} \times \mathcal{E})^*$ the policy π attains the optimal value: $V_\nu^\pi(æ_{<t}) = V_\nu^*(æ_{<t})$.

We assumed that the discount function is summable, rewards are bounded (Assumption 17a), and actions and percepts spaces are both finite (Assumption 17b). Therefore an optimal deterministic policy exists for every environment [LH14, Thm. 10].

3.2 REFLECTIVE BAYESIAN AGENTS

Fix O to be a reflective oracle. From now on, we assume that the action space $\mathcal{A} := \{\alpha, \beta\}$ is binary. We can treat computable measures over binary strings as environments: the environment ν corresponding to a probabilistic Turing machine $T \in \mathcal{T}$ is defined by

$$\nu(e_t \mid æ_{<t}a_t) := \overline{\lambda}_T^O(y \mid x) = \prod_{i=1}^{k} \overline{\lambda}_T^O(y_i \mid xy_1 \ldots y_{i-1})$$

where $y_{1:k}$ is a binary encoding of e_t and x is a binary encoding of $æ_{<t}a_t$. The actions $a_{1:\infty}$ are only *contextual*, and not part of the environment distribution. We define

$$\nu(e_{<t} \mid a_{<t}) := \prod_{k=1}^{t-1} \nu(e_k \mid æ_{<k}).$$

Let $T_1, T_2, \ldots$ be an enumeration of all probabilistic Turing machines in $\mathcal{T}$. We define the *class of reflective environments*

$$\mathcal{M}_{\text{refl}}^O := \left\{\overline{\lambda}_{T_1}^O, \overline{\lambda}_{T_2}^O, \ldots\right\}.$$

This is the class of all environments computable on a probabilistic Turing machine with reflective oracle O, that have been completed from semimeasures to measures using O.

Analogously to AIXI [Hut05], we define a Bayesian mixture over the class $\mathcal{M}_{\text{refl}}^O$. Let $w \in \Delta\mathcal{M}_{\text{refl}}^O$ be a lower semicomputable prior probability distribution on $\mathcal{M}_{\text{refl}}^O$. Possible choices for the prior include the *Solomonoff prior* $w(\overline{\lambda}_T^O) := 2^{-K(T)}$, where $K(T)$ denotes the length of the shortest input to some universal Turing machine that encodes T [Sol78].[2] We define the corresponding Bayesian mixture

$$\xi(e_t \mid æ_{<t}a_t) := \sum_{\nu \in \mathcal{M}_{\text{refl}}^O} w(\nu \mid æ_{<t}) \nu(e_t \mid æ_{<t}a_t) \tag{3}$$

where $w(\nu \mid æ_{<t})$ is the (renomalized) posterior,

$$w(\nu \mid æ_{<t}) := w(\nu) \frac{\nu(e_{<t} \mid a_{<t})}{\overline{\xi}(e_{<t} \mid a_{<t})}. \tag{4}$$

The mixture ξ is lower semicomputable on an oracle Turing machine because the posterior $w(\cdot \mid æ_{<t})$ is lower semicomputable. Hence there is an oracle machine T such that $\xi = \lambda_T^O$. We define its completion $\overline{\xi} := \overline{\lambda}_T^O$ as the completion of λ_T^O. This is the distribution that is used to compute the posterior. There are no cyclic dependencies since $\overline{\xi}$ is called on the shorter history $æ_{<t}$. We arrive at the following statement.

Proposition 20 (Bayes is in the Class)**.** $\overline{\xi} \in \mathcal{M}_{\text{refl}}^O$.

Moreover, since O is reflective, we have that $\overline{\xi}$ dominates all environments $\nu \in \mathcal{M}_{\text{refl}}^O$:

$$\overline{\xi}(e_{1:t} \mid a_{1:t})$$

[2] Technically, the lower semicomputable prior $2^{-K(T)}$ is only a semidistribution because it does not sum to 1. This turns out to be unimportant.

$$
\begin{aligned}
&= \overline{\xi}(e_t \mid \text{æ}_{<t}a_t)\overline{\xi}(e_{<t} \mid a_{<t}) \\
&\geq \xi(e_t \mid \text{æ}_{<t}a_t)\overline{\xi}(e_{<t} \mid a_{<t}) \\
&= \overline{\xi}(e_{<t} \mid a_{<t}) \sum_{\nu \in \mathcal{M}^O_{\text{refl}}} w(\nu \mid \text{æ}_{<t})\nu(e_t \mid \text{æ}_{<t}a_t) \\
&= \overline{\xi}(e_{<t} \mid a_{<t}) \sum_{\nu \in \mathcal{M}^O_{\text{refl}}} w(\nu)\frac{\nu(e_{<t} \mid a_{<t})}{\overline{\xi}(e_{<t} \mid a_{<t})}\nu(e_t \mid \text{æ}_{<t}a_t) \\
&= \sum_{\nu \in \mathcal{M}^O_{\text{refl}}} w(\nu)\nu(e_{1:t} \mid a_{1:t}) \\
&\geq w(\nu)\nu(e_{1:t} \mid a_{1:t})
\end{aligned}
$$

This property is crucial for on-policy value convergence.

Lemma 21 (On-Policy Value Convergence [Hut05, Thm. 5.36])**.** *For any policy π and any environment $\mu \in \mathcal{M}^O_{\text{refl}}$ with $w(\mu) > 0$,*

$$V^\pi_\mu(\text{æ}_{<t}) - V^\pi_{\overline{\xi}}(\text{æ}_{<t}) \to 0 \ \mu^\pi\text{-almost surely as } t \to \infty.$$

3.3 REFLECTIVE-ORACLE-COMPUTABLE POLICIES

This subsection is dedicated to the following result that was previously stated but not proved in [FST15, Alg. 6]. It contrasts results on arbitrary semicomputable environments where optimal policies are not limit computable [LH15b, Sec. 4].

Theorem 22 (Optimal Policies are Oracle Computable)**.** *For every $\nu \in \mathcal{M}^O_{\text{refl}}$, there is a ν-optimal (stochastic) policy π^*_ν that is reflective-oracle-computable.*

Note that even though deterministic optimal policies always exist, those policies are typically not reflective-oracle-computable.

To prove Theorem 22 we need the following lemma.

Lemma 23 (Reflective-Oracle-Computable Optimal Value Function)**.** *For every environment $\nu \in \mathcal{M}^O_{\text{refl}}$ the optimal value function V^*_ν is reflective-oracle-computable.*

Proof. This proof follows the proof of [LH15b, Cor. 13]. We write the optimal value explicitly as

$$V^*_\nu(\text{æ}_{<t}) = \frac{1}{\Gamma_t} \lim_{m\to\infty} \sum_{\text{æ}_{t:m}} \max \sum_{k=t}^{m} \gamma_k r_k \prod_{i=t}^{k} \nu(e_i \mid \text{æ}_{<i}), \tag{5}$$

where $\sum\max$ denotes the expectimax operator:

$$\sum_{\text{æ}_{t:m}} \max := \max_{a_t \in \mathcal{A}} \sum_{e_t \in \mathcal{E}} \dots \max_{a_m \in \mathcal{A}} \sum_{e_m \in \mathcal{E}}$$

For a fixed m, all involved quantities are reflective-oracle-computable. Moreover, this quantity is monotone increasing in m and the tail sum from $m+1$ to ∞ is bounded by Γ_{m+1} which is computable according to Assumption 17c and converges to 0 as $m \to \infty$. Therefore we can enumerate all rationals above and below V^*_ν. □

Proof of Theorem 22. According to Lemma 23 the optimal value function V^*_ν is reflective-oracle-computable. Hence there is a probabilistic Turing machine T such that

$$\lambda^O_T(1 \mid \text{æ}_{<t}) = \big(V^*_\nu(\text{æ}_{<t}\alpha) - V^*_\nu(\text{æ}_{<t}\beta) + 1\big)/2.$$

We define a policy π that takes action α if $O(T, \text{æ}_{<t}, 1/2) = 1$ and action β if $O(T, \text{æ}_{<t}, 1/2) = 0$. (This policy is stochastic because the answer of the oracle O is stochastic.)

It remains to show that π is a ν-optimal policy. If $V^*_\nu(\text{æ}_{<t}\alpha) > V^*_\nu(\text{æ}_{<t}\beta)$, then $\lambda^O_T(1 \mid \text{æ}_{<t}) > 1/2$, thus $O(T, \text{æ}_{<t}, 1/2) = 1$ since O is reflective, and hence π takes action α. Conversely, if $V^*_\nu(\text{æ}_{<t}\alpha) < V^*_\nu(\text{æ}_{<t}\beta)$, then $\lambda^O_T(1 \mid \text{æ}_{<t}) < 1/2$, thus $O(T, \text{æ}_{<t}, 1/2) = 0$ since O is reflective, and hence π takes action β. Lastly, if $V^*_\nu(\text{æ}_{<t}\alpha) = V^*_\nu(\text{æ}_{<t}\beta)$, then both actions are optimal and thus it does not matter which action is returned by policy π. (This is the case where the oracle may randomize.) □

3.4 SOLUTION TO THE GRAIN OF TRUTH PROBLEM

Together, Proposition 20 and Theorem 22 provide the necessary ingredients to solve the grain of truth problem.

Corollary 24 (Solution to the Grain of Truth Problem)**.** *For every lower semicomputable prior $w \in \Delta\mathcal{M}^O_{\text{refl}}$ the Bayes-optimal policy $\pi^*_{\overline{\xi}}$ is reflective-oracle-computable where ξ is the Bayes-mixture corresponding to w defined in* (3)*.*

Proof. From Proposition 20 and Theorem 22. □

Hence the environment class $\mathcal{M}^O_{\text{refl}}$ contains any reflective-oracle-computable modification of the Bayes-optimal policy $\pi^*_{\overline{\xi}}$. In particular, this includes computable multi-agent environments that contain other Bayesian agents over the class $\mathcal{M}^O_{\text{refl}}$. So any Bayesian agent over the class $\mathcal{M}^O_{\text{refl}}$ has a grain of truth even though the environment may contain other Bayesian agents *of equal power*. We proceed to sketch the implications for multi-agent environments in the next section.

4 MULTI-AGENT ENVIRONMENTS

This section summarizes our results for multi-agent systems. The proofs can be found in [Lei16].

4.1 SETUP

In a *multi-agent environment* there are n agents each taking sequential actions from the finite action space $\mathcal{A}$. In each time step $t = 1, 2, \dots$, the environment receives action a^i_t from agent i and outputs n percepts $e^1_t, \dots, e^n_t \in \mathcal{E}$, one for

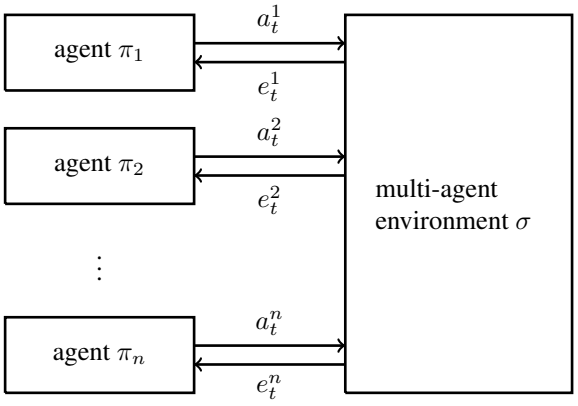

Figure 2: Agents $\pi_1, \ldots, \pi_n$ interacting in a multi-agent environment.

each agent. Each percept $e_t^i = (o_t^i, r_t^i)$ contains an observation o_t^i and a reward $r_t^i \in [0, 1]$. Importantly, agent i only sees its own action a_t^i and its own percept e_t^i (see Figure 2). We use the shorthand notation $a_t := (a_t^1, \ldots, a_t^n)$ and $e_t := (e_t^1, \ldots, e_t^n)$ and denote $\text{æ}_{<t}^i = a_1^i e_1^i \ldots a_{t-1}^i e_{t-1}^i$ and $\text{æ}_{<t} = a_1 e_1 \ldots a_{t-1} e_{t-1}$.

We define a multi-agent environment as a function

$$\sigma : (\mathcal{A}^n \times \mathcal{E}^n)^* \times \mathcal{A}^n \to \Delta(\mathcal{E}^n).$$

The agents are given by n policies $\pi_1, \ldots, \pi_n$ where $\pi_i : (\mathcal{A} \times \mathcal{E})^* \to \Delta\mathcal{A}$. Together they specify the *history distribution*

$$\begin{aligned} \sigma^{\pi_{1:n}}(\epsilon) &:= 1 \\ \sigma^{\pi_{1:n}}(\text{æ}_{1:t}) &:= \sigma^{\pi_{1:n}}(\text{æ}_{<t} a_t) \sigma(e_t \mid \text{æ}_{<t} a_t) \\ \sigma^{\pi_{1:n}}(\text{æ}_{<t} a_t) &:= \sigma^{\pi_{1:n}}(\text{æ}_{<t}) \prod_{i=1}^{n} \pi_i(a_t^i \mid \text{æ}_{<t}^i). \end{aligned}$$

Each agent i acts in a *subjective environment* σ_i given by joining the multi-agent environment σ with the policies $\pi_1, \ldots, \pi_{i-1}, \pi_{i+1}, \ldots, \pi_n$ by marginalizing over the histories that π_i does not see. Together with policy π_i, the environment σ_i yields a distribution over the histories of agent i

$$\sigma_i^{\pi_i}(\text{æ}_{<t}^i) := \sum_{\text{æ}_{<t}^j, j \neq i} \sigma^{\pi_{1:n}}(\text{æ}_{<t}).$$

We get the definition of the subjective environment σ_i with the identity $\sigma_i(e_t^i \mid \text{æ}_{<t}^i a_t^i) := \sigma_i^{\pi_i}(e_t^i \mid \text{æ}_{<t}^i a_t^i)$. It is crucial to note that the subjective environment σ_i and the policy π_i are ordinary environments and policies, so we can use the formalism from Section 3.

Our definition of a multi-agent environment is very general and encompasses most of game theory. It allows for cooperative, competitive, and mixed games; infinitely repeated games or any (infinite-length) extensive form games with finitely many players.

The policy π_i is an *ε-best response* after history $\text{æ}_{<t}^i$ iff

$$V_{\sigma_i}^*(\text{æ}_{<t}^i) - V_{\sigma_i}^{\pi_i}(\text{æ}_{<t}^i) < \varepsilon.$$

If at some time step t, all agents' policies are ε-best responses, we have an *ε-Nash equilibrium*. The property of multi-agent systems that is analogous to asymptotic optimality is convergence to an ε-Nash equilibrium.

4.2 INFORMED REFLECTIVE AGENTS

Let σ be a multi-agent environment and let $\pi_{\sigma_1}^*, \ldots \pi_{\sigma_n}^*$ be such that for each i the policy $\pi_{\sigma_i}^*$ is an optimal policy in agent i's subjective environment σ_i. At first glance this seems ill-defined: The subjective environment σ_i depends on each other policy $\pi_{\sigma_j}^*$ for $j \neq i$, which depends on the subjective environment σ_j, which in turn depends on the policy $\pi_{\sigma_i}^*$. However, this circular definition actually has a well-defined solution.

Theorem 25 (Optimal Multi-Agent Policies)**.** *For any reflective-oracle-computable multi-agent environment σ, the optimal policies $\pi_{\sigma_1}^*, \ldots, \pi_{\sigma_n}^*$ exist and are reflective-oracle-computable.*

Note the strength of Theorem 25: each of the policies $\pi_{\sigma_i}^*$ is acting optimally *given the knowledge of everyone else's policies*. Hence optimal policies play 0-best responses by definition, so if every agent is playing an optimal policy, we have a Nash equilibrium. Moreover, this Nash equilibrium is also a *subgame perfect* Nash equilibrium, because each agent also acts optimally on the counterfactual histories that do not end up being played. In other words, Theorem 25 states the existence and reflective-oracle-computability of a subgame perfect Nash equilibrium in any reflective-oracle-computable multi-agent environment. From Theorem 6 we then get that these subgame perfect Nash equilibria are limit computable.

Corollary 26 (Solution to Computable Multi-Agent Environments)**.** *For any computable multi-agent environment σ, the optimal policies $\pi_{\sigma_1}^*, \ldots, \pi_{\sigma_n}^*$ exist and are limit computable.*

4.3 LEARNING REFLECTIVE AGENTS

Since our class $\mathcal{M}_{\text{refl}}^O$ solves the grain of truth problem, the result by Kalai and Lehrer [KL93] immediately implies that for any Bayesian agents $\pi_1, \ldots, \pi_n$ interacting in an infinitely repeated game and for all $\varepsilon > 0$ and all $i \in \{1, \ldots, n\}$ there is almost surely a $t_0 \in \mathbb{N}$ such that for all $t \geq t_0$ the policy π_i is an ε-best response. However, this hinges on the important fact that every agent has to know the game and also that all other agents are Bayesian agents. Otherwise the convergence to an ε-Nash equilibrium may fail, as illustrated by the following example.

At the core of the following construction is a *dogmatic prior* [LH15a, Sec. 3.2]. A dogmatic prior assigns very

high probability to going to hell (reward 0 forever) if the agent deviates from a given computable policy π. For a Bayesian agent it is thus only worth deviating from the policy π if the agent thinks that the prospects of following π are very poor already. This implies that for general multi-agent environments and without additional assumptions on the prior, we cannot prove any meaningful convergence result about Bayesian agents acting in an unknown multi-agent environment.

Example 27 (Reflective Bayesians Playing Matching Pennies)**.** In the game of *matching pennies* there are two agents ($n = 2$), and two actions $\mathcal{A} = \{\alpha, \beta\}$ representing the two sides of a penny. In each time step agent 1 wins if the two actions are identical and agent 2 wins if the two actions are different. The payoff matrix is as follows.

	α	β
α	1,0	0,1
β	0,1	1,0

We use $\mathcal{E} = \{0, 1\}$ to be the set of rewards (observations are vacuous) and define the multi-agent environment σ to give reward 1 to agent 1 iff $a_t^1 = a_t^2$ (0 otherwise) and reward 1 to agent 2 iff $a_t^1 \neq a_t^2$ (0 otherwise). Note that neither agent knows a priori that they are playing matching pennies, nor that they are playing an infinite repeated game with one other player.

Let π_1 be the policy that takes the action sequence $(\alpha\alpha\beta)^\infty$ and let $\pi_2 := \pi_\alpha$ be the policy that always takes action α. The average reward of policy π_1 is $2/3$ and the average reward of policy π_2 is $1/3$. Let ξ be a universal mixture (3). By Lemma 21, $V_\xi^{\pi_1} \to c_1 \approx 2/3$ and $V_\xi^{\pi_2} \to c_2 \approx 1/3$ almost surely when following policies (π_1, π_2). Therefore there is an $\varepsilon > 0$ such that $V_\xi^{\pi_1} > \varepsilon$ and $V_\xi^{\pi_2} > \varepsilon$ for all time steps. Now we can apply [LH15a, Thm. 7] to conclude that there are (dogmatic) mixtures ξ_1' and ξ_2' such that $\pi^*_{\xi_1'}$ always follows policy π_1 and $\pi^*_{\xi_2'}$ always follows policy π_2. This does not converge to a (ε-)Nash equilibrium. $\lozenge$

A policy π is *asymptotically optimal in mean in an environment class* $\mathcal{M}$ iff for all $\mu \in \mathcal{M}$

$$\mathbb{E}_\mu^\pi\big[V_\mu^*(\text{æ}_{<t}) - V_\mu^\pi(\text{æ}_{<t})\big] \to 0 \text{ as } t \to \infty \qquad (6)$$

where $\mathbb{E}_\mu^\pi$ denotes the expectation with respect to the probability distribution μ^π over histories generated by policy π acting in environment μ.

Asymptotic optimality stands out because it is currently the only known nontrivial objective notion of optimality in general reinforcement learning [LH15a].

The following theorem is the main convergence result. It states that for asymptotically optimal agents we get convergence to ε-Nash equilibria in any reflective-oracle-computable multi-agent environment.

Theorem 28 (Convergence to Equilibrium)**.** *Let σ be an reflective-oracle-computable multi-agent environment and let $\pi_1, \ldots, \pi_n$ be reflective-oracle-computable policies that are asymptotically optimal in mean in the class $\mathcal{M}^O_{\text{refl}}$. Then for all $\varepsilon > 0$ and all $i \in \{1, \ldots, n\}$ the $\sigma^{\pi_{1:n}}$-probability that the policy π_i is an ε-best response converges to 1 as $t \to \infty$.*

In contrast to Theorem 25 which yields policies that play a subgame perfect equilibrium, this is not the case for Theorem 28: the agents typically do not learn to predict off-policy and thus will generally not play ε-best responses in the counterfactual histories that they never see. This weaker form of equilibrium is unavoidable if the agents do not know the environment because it is impossible to learn the parts that they do not interact with.

Together with Theorem 6 and the asymptotic optimality of the Thompson sampling policy [LLOH16, Thm. 4] that is reflective-oracle computable we get the following corollary.

Corollary 29 (Convergence to Equilibrium)**.** *There are limit computable policies $\pi_1, \ldots, \pi_n$ such that for any computable multi-agent environment σ and for all $\varepsilon > 0$ and all $i \in \{1, \ldots, n\}$ the $\sigma^{\pi_{1:n}}$-probability that the policy π_i is an ε-best response converges to 1 as $t \to \infty$.*

5 DISCUSSION

This paper introduced the class of all reflective-oracle-computable environments $\mathcal{M}^O_{\text{refl}}$. This class solves the grain of truth problem because it contains (any computable modification of) Bayesian agents defined over $\mathcal{M}^O_{\text{refl}}$: the optimal agents and Bayes-optimal agents over the class are all reflective-oracle-computable (Theorem 22 and Corollary 24).

If the environment is unknown, then a Bayesian agent may end up playing suboptimally (Example 27). However, if each agent uses a policy that is asymptotically optimal in mean (such as the Thompson sampling policy [LLOH16]) then for every $\varepsilon > 0$ the agents converge to an ε-Nash equilibrium (Theorem 28 and Corollary 29).

Our solution to the grain of truth problem is purely theoretical. However, Theorem 6 shows that our class $\mathcal{M}^O_{\text{refl}}$ allows for computable approximations. This suggests that practical approaches can be derived from this result, and reflective oracles have already seen applications in one-shot games [FTC15b].

Acknowledgements

We thank Marcus Hutter and Tom Everitt for valuable comments.

REFERENCES

[FST15] Benja Fallenstein, Nate Soares, and Jessica Taylor. Reflective variants of Solomonoff induction and AIXI. In *Artificial General Intelligence*. Springer, 2015.

[FTC15a] Benja Fallenstein, Jessica Taylor, and Paul F Christiano. Reflective oracles: A foundation for classical game theory. Technical report, Machine Intelligence Research Institute, 2015. `http://arxiv.org/abs/1508.04145`.

[FTC15b] Benja Fallenstein, Jessica Taylor, and Paul F Christiano. Reflective oracles: A foundation for game theory in artificial intelligence. In *Logic, Rationality, and Interaction*, pages 411–415. Springer, 2015.

[FY01] Dean P Foster and H Peyton Young. On the impossibility of predicting the behavior of rational agents. *Proceedings of the National Academy of Sciences*, 98(22):12848–12853, 2001.

[Hut05] Marcus Hutter. *Universal Artificial Intelligence*. Springer, 2005.

[Hut09] Marcus Hutter. Open problems in universal induction & intelligence. *Algorithms*, 3(2):879–906, 2009.

[KL93] Ehud Kalai and Ehud Lehrer. Rational learning leads to Nash equilibrium. *Econometrica*, pages 1019–1045, 1993.

[Kle52] Stephen Cole Kleene. *Introduction to Metamathematics*. Wolters-Noordhoff Publishing, 1952.

[Lei16] Jan Leike. *Nonparametric General Reinforcement Learning*. PhD thesis, Australian National University, 2016.

[LH14] Tor Lattimore and Marcus Hutter. General time consistent discounting. *Theoretical Computer Science*, 519:140–154, 2014.

[LH15a] Jan Leike and Marcus Hutter. Bad universal priors and notions of optimality. In *Conference on Learning Theory*, pages 1244–1259, 2015.

[LH15b] Jan Leike and Marcus Hutter. On the computability of AIXI. In *Uncertainty in Artificial Intelligence*, pages 464–473, 2015.

[LH15c] Jan Leike and Marcus Hutter. On the computability of Solomonoff induction and knowledge-seeking. In *Algorithmic Learning Theory*, pages 364–378, 2015.

[LLOH16] Jan Leike, Tor Lattimore, Laurent Orseau, and Marcus Hutter. Thompson sampling is asymptotically optimal in general environments. In *Uncertainty in Artificial Intelligence*, 2016.

[LV08] Ming Li and Paul M. B. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications*. Texts in Computer Science. Springer, 3rd edition, 2008.

[Nac97] John H Nachbar. Prediction, optimization, and learning in repeated games. *Econometrica*, 65(2):275–309, 1997.

[Nac05] John H Nachbar. Beliefs in repeated games. *Econometrica*, 73(2):459–480, 2005.

[Ors13] Laurent Orseau. Asymptotic non-learnability of universal agents with computable horizon functions. *Theoretical Computer Science*, 473:149–156, 2013.

[SLB09] Yoav Shoham and Kevin Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, 2009.

[Sol78] Ray Solomonoff. Complexity-based induction systems: Comparisons and convergence theorems. *IEEE Transactions on Information Theory*, 24(4):422–432, 1978.