On the Computability of Solomonoff Induction and Knowledge-Seeking

Jan Leike(✉) and Marcus Hutter

Australian National University, Canberra, Australia
{jan.leike,marcus.hutter}@anu.edu.au

Abstract. Solomonoff induction is held as a gold standard for learning, but it is known to be incomputable. We quantify its incomputability by placing various flavors of Solomonoff's prior M in the arithmetical hierarchy. We also derive computability bounds for knowledge-seeking agents, and give a limit-computable weakly asymptotically optimal reinforcement learning agent.

Keywords: Solomonoff induction · Exploration · Knowledge-seeking agents · General reinforcement learning · Asymptotic optimality · Computability · Complexity · Arithmetical hierarchy · Universal turing machine · AIXI · BayesExp

1 Introduction

Solomonoff's theory of learning [11,19,20], commonly called *Solomonoff induction*, arguably solves the induction problem [18]: for data drawn from any computable measure μ, Solomonoff induction will converge to the correct belief about any hypothesis [1]. Moreover, convergence is extremely fast in the sense that the expected number of prediction errors is $E + O(\sqrt{E})$ compared to the number of errors E made by the informed predictor that knows μ [4].

In *reinforcement learning* an agent repeatedly takes actions and receives observations and rewards. The goal is to maximize cumulative (discounted) reward. Solomonoff's ideas can be extended to reinforcement learning, leading to the Bayesian agent AIXI [3,5]. However, AIXI's trade-off between exploration and exploitation includes insufficient exploration to get rid of the prior's bias [9], which is why the universal agent AIXI does not achieve asymptotic optimality [13,15].

For extra exploration, we can resort to Orseau's *knowledge-seeking agents*. Instead of rewards, knowledge-seeking agents maximize entropy gain [14,16] or expected information gain [17]. These agents are apt explorers, and asymptotically they learn their environment perfectly [16,17].

A reinforcement learning agent is *weakly asymptotically optimal* if the value of its policy converges to the optimal value in Cesro mean [7]. Weak asymptotic optimality stands out because it currently is the only known nontrivial objective notion of optimality for general reinforcement learners [7,9,15]. Lattimore defines

K. Chaudhuri et al. (Eds.): ALT 2015, LNAI 9355, pp. 364–378, 2015.
DOI: 10.1007/978-3-319-24486-0_24

Table 1. The computability results on M, M_{norm}, $\overline{M}$, and $\overline{M}_{\text{norm}}$ proved in Section 3. Lower bounds on the complexity of $\overline{M}$ and $\overline{M}_{\text{norm}}$ are given only for specific universal Turing machines.

P	$\{(x,q) \in \mathcal{X}^* \times \mathbb{Q} \mid P(x) > q\}$	$\{(x,y,q) \in \mathcal{X}^* \times \mathcal{X}^* \times \mathbb{Q} \mid P(xy \mid x) > q\}$
M	$\Sigma_1^0 \setminus \Delta_1^0$	$\Delta_2^0 \setminus (\Sigma_1^0 \cup \Pi_1^0)$
M_{norm}	$\Delta_2^0 \setminus (\Sigma_1^0 \cup \Pi_1^0)$	$\Delta_2^0 \setminus (\Sigma_1^0 \cup \Pi_1^0)$
$\overline{M}$	$\Pi_2^0 \setminus \Delta_2^0$	$\Delta_3^0 \setminus (\Sigma_2^0 \cup \Pi_2^0)$
$\overline{M}_{\text{norm}}$	$\Delta_3^0 \setminus (\Sigma_2^0 \cup \Pi_2^0)$	$\Delta_3^0 \setminus (\Sigma_2^0 \cup \Pi_2^0)$

the agent BayesExp by grafting a knowledge-seeking component on top of AIXI and shows that BayesExp is a weakly asymptotically optimal agent in the class of all stochastically computable environments [6, Ch. 5].

The purpose of models such as Solomonoff induction, AIXI, and knowledge-seeking agents is to answer the question of how to solve (reinforcement) learning *in theory*. These answers are useless if they cannot be approximated in practice, i.e., by a regular Turing machine. Therefore we posit that any ideal model must at least be *limit computable* (Δ_2^0).

Limit computable functions are the functions that admit an *anytime algorithm*. More generally, the *arithmetical hierarchy* specifies different levels of computability based on *oracle machines*: each level in the arithmetical hierarchy is computed by a Turing machine which may query a halting oracle for the respective lower level.

In previous work [10] we established that AIXI is limit computable if restricted to ε-optimal policies, and placed various versions of AIXI, AINU, and AIMU in the arithmetical hierarchy. In this paper we investigate the (in-)computability of Solomonoff induction and knowledge-seeking. The universal prior M is lower semicomputable and hence its conditional is limit computable. But M is a semimeasure: it assigns positive probability that the observed string has only finite length. This can be circumvented by normalizing M. Solomonoff's normalization M_{norm} preserves the ratio $M(x1)/M(x0)$ and is limit computable. If we remove the contribution of programs that compute only finite strings, we get a semimeasure $\overline{M}$, which can be normalized to $\overline{M}_{\text{norm}}$ by multiplication with a constant. We show that both $\overline{M}$ and $\overline{M}_{\text{norm}}$ are *not* limit computable. Our results on the computability of Solomonoff induction are stated in Table 1 and proved in Section 3. In Section 4 we show that for finite horizons both the entropy-seeking and the information-seeking agent are Δ_3^0-computable and have limit-computable ε-optimal policies. The weakly asymptotically optimal agent BayesExp relies on optimal policies that are generally not limit computable [10, Thm.16]. In Section 5 we give a weakly asymptotically optimal agent based on BayesExp that is limit computable. A list of notation can be found on page 377.

2 Preliminaries

We use the setup and notation from [10].

2.1 The Arithmetical Hierarchy

A set $A \subseteq \mathbb{N}$ *is* Σ^0_n iff there is a computable relation S such that

$$k \in A \iff \exists k_1 \forall k_2 \ldots Q_n k_n \; S(k, k_1, \ldots, k_n) \tag{1}$$

where $Q_n = \forall$ if n is even, $Q_n = \exists$ if n is odd [12, Def. 1.4.10]. A set $A \subseteq \mathbb{N}$ *is* Π^0_n iff its complement $\mathbb{N} \setminus A$ is Σ^0_n. We call the formula on the right hand side of (1) a Σ^0_n*-formula*, its negation is called Π^0_n*-formula*. It can be shown that we can add any bounded quantifiers and duplicate quantifiers of the same type without changing the classification of A. The set A *is* Δ^0_n iff A is Σ^0_n and A is Π^0_n. We get that Σ^0_1 as the class of recursively enumerable sets, Π^0_1 as the class of co-recursively enumerable sets and Δ^0_1 as the class of recursive sets.

We say the set $A \subseteq \mathbb{N}$ is Σ^0_n*-hard* (Π^0_n*-hard,* Δ^0_n*-hard)* iff for any set $B \in \Sigma^0_n$ ($B \in \Pi^0_n$, $B \in \Delta^0_n$), B is many-one reducible to A, i.e., there is a computable function f such that $k \in B \leftrightarrow f(k) \in A$ [12, Def. 1.2.1]. We get $\Sigma^0_n \subset \Delta^0_{n+1} \subset \Sigma^0_{n+1} \subset \ldots$ and $\Pi^0_n \subset \Delta^0_{n+1} \subset \Pi^0_{n+1} \subset \ldots$. This hierarchy of subsets of natural numbers is known as the *arithmetical hierarchy*.

By Post's Theorem [12, Thm. 1.4.13], a set is Σ^0_n if and only if it is recursively enumerable on an oracle machine with an oracle for a Σ^0_{n-1}-hard set.

2.2 Strings

Let $\mathcal{X}$ be some 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 $x \in \mathcal{X}^*$, 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. We write $x \sqsubseteq y$ iff x is a prefix of y, i.e., $x = y_{1:|x|}$.

2.3 Computability of Real-Valued Functions

We fix some encoding of rational numbers into binary strings and an encoding of binary strings into natural numbers. From now on, this encoding will be done implicitly wherever necessary.

Definition 1 (Σ^0_n-, Π^0_n-, Δ^0_n-computable). *A function $f : \mathcal{X}^* \to \mathbb{R}$ is called* Σ^0_n-computable (Π^0_n-computable, Δ^0_n-computable) *iff the set* $\{(x, q) \in \mathcal{X}^* \times \mathbb{Q} \mid f(x) > q\}$ *is* Σ^0_n *(Π^0_n, Δ^0_n).*

Table 2. Connection between the computability of real-valued functions and the arithmetical hierarchy.

	$\{(x,q) \mid f(x) > q\}$	$\{(x,q) \mid f(x) < q\}$
f is computable	Δ_1^0	Δ_1^0
f is lower semicomputable	Σ_1^0	Π_1^0
f is upper semicomputable	Π_1^0	Σ_1^0
f is limit computable	Δ_2^0	Δ_2^0
f is Δ_n^0-computable	Δ_n^0	Δ_n^0
f is Σ_n^0-computable	Σ_n^0	Π_n^0
f is Π_n^0-computable	Π_n^0	Σ_n^0

A Δ_1^0-computable function is called *computable*, a Σ_1^0-computable function is called *lower semicomputable*, and a Π_1^0-computable function is called *upper semicomputable*. A Δ_2^0-computable function f is called *limit computable*, because 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.

Limit-computable sets are the highest level in the arithmetical hierarchy that can be approached by a regular Turing machine. Above limit-computable sets we necessarily need some form of halting oracle. See Table 2 for the definition of lower/upper semicomputable and limit-computable functions in terms of the arithmetical hierarchy.

Lemma 2 (Computability of Arithmetical Operations). *Let $n > 0$ and let $f, g : \mathcal{X}^* \to \mathbb{R}$ be two Δ_n^0-computable functions. Then*

(i) $\{(x,y) \mid f(x) > g(y)\}$ *is* Σ_n^0,
(ii) $\{(x,y) \mid f(x) \leq g(y)\}$ *is* Π_n^0,
(iii) $f+g$, $f-g$, *and* $f \cdot g$ *are* Δ_n^0*-computable,*
(iv) f/g *is* Δ_n^0*-computable if* $g(x) \neq 0$ *for all* x, *and*
(v) $\log f$ *is* Δ_n^0*-computable if* $f(x) > 0$ *for all* x.

3 The Complexity of Solomonoff Induction

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 $x \in \mathcal{X}^*$. A semimeasure is called (probability) *measure* iff for all x equalities hold in (i) and (ii).

$$M(xy \mid x) > q \iff \forall \ell \exists k \, \frac{\phi(xy,k)}{\phi(x,\ell)} > q \iff \exists k \exists \ell_0 \forall \ell \geq \ell_0 \, \frac{\phi(xy,k)}{\phi(x,\ell)} > q$$

Fig. 1. A Π_2^0-formula and an equivalent Σ_2^0-formula defining conditional M. Here $\phi(x,k)$ denotes a computable function that lower semicomputes $M(x)$.

Solomonoff's prior M [19] assigns to a string x the probability that the reference universal monotone Turing machine U [11, Ch. 4.5.2] computes a string starting with x when fed with uniformly random bits as input. Formally,

$$M(x) := \sum_{p:\, x \sqsubseteq U(p)} 2^{-|p|}. \tag{2}$$

The function M is a lower semicomputable semimeasure, but not computable and not a measure [11, Lem. 4.5.3]. A semimeasure ν can be turned into a measure ν_{norm} using *Solomonoff normalization*: $\nu_{\text{norm}}(\epsilon) := 1$ and for all $x \in \mathcal{X}^*$ and $a \in \mathcal{X}$,

$$\nu_{\text{norm}}(xa) := \nu_{\text{norm}}(x) \frac{\nu(xa)}{\sum_{b \in \mathcal{X}} \nu(xb)}. \tag{3}$$

By definition, M_{norm} and $\overline{M}_{\text{norm}}$ are measures [11, Sec. 4.5.3]. Moreover, since $M_{\text{norm}} \geq M$, normalization preserves universal dominance. Hence Solomonoff's theorem implies that M_{norm} predicts just as well as M.

The *measure mixture* $\overline{M}$ [2, p. 74] is defined as

$$\overline{M}(x) := \lim_{n \to \infty} \sum_{y \in \mathcal{X}^n} M(xy). \tag{4}$$

The measure mixture $\overline{M}$ is the same as M except that the contributions by programs that do not produce infinite strings are removed: for any such program p, let k denote the length of the finite string generated by p. Then for $|xy| > k$, the program p does not contribute to $M(xy)$, hence it is excluded from $\overline{M}(x)$.

Similarly to M, the measure mixture $\overline{M}$ is not a (probability) measure since $\overline{M}(\varepsilon) < 1$, but in this case normalization (3) is just multiplication with the constant $1/\overline{M}(\epsilon)$, leading to the *normalized measure mixture* $\overline{M}_{\text{norm}}$. When using the Solomonoff prior M (or one of its sisters M_{norm}, $\overline{M}$, or $\overline{M}_{\text{norm}}$) for sequence prediction, we need to compute the conditional probability $M(xy \mid x) := M(xy)/M(x)$ for finite strings $x, y \in \mathcal{X}^*$. Because $M(x) > 0$ for all finite strings $x \in \mathcal{X}^*$, this quotient is well-defined.

Theorem 3 (Complexity of M, M_{norm}, $\overline{M}$, and $\overline{M}_{\text{norm}}$).

(i) $M(x)$ *is lower semicomputable*
(ii) $M(xy \mid x)$ *is limit computable*
(iii) $M_{\text{norm}}(x)$ *is limit computable*
(iv) $M_{\text{norm}}(xy \mid x)$ *is limit computable*
(v) $\overline{M}(x)$ *is* Π_2^0*-computable*
(vi) $\overline{M}(xy \mid x)$ *is* Δ_3^0*-computable*
(vii) $\overline{M}_{\text{norm}}(x)$ *is* Δ_3^0*-computable*
(viii) $\overline{M}_{\text{norm}}(xy \mid x)$ *is* Δ_3^0*-computable*

Proof. (i) By [11, Thm. 4.5.2]. Intuitively, we can run all programs in parallel and get monotonely increasing lower bounds for $M(x)$ by adding $2^{-|p|}$ every time a program p has completed outputting x.
(ii) From (i) and Lemma 2 (iv), since $M(x) > 0$ (see also Figure 1).
(iii) By Lemma 2 (iii,iv) and $M(x) > 0$.
(iv) By (iii) and Lemma 2 (iv), since $M_{\text{norm}}(x) \geq M(x) > 0$.
(v) Let ϕ be a computable function that lower semicomputes M. Since M is a semimeasure, $M(xy) \geq \sum_z M(xyz)$, hence $\sum_{y \in \mathcal{X}^n} M(xy)$ is nonincreasing in n and thus $\overline{M}(x) > q$ iff $\forall n \exists k \sum_{y \in \mathcal{X}^n} \phi(xy, k) > q$.
(vi) From (v) and Lemma 2 (iv), since $\overline{M}(x) > 0$.
(vii) From (v) and Lemma 2 (iv).
(viii) From (vi) and Lemma 2 (iv), since $\overline{M}_{\text{norm}}(x) \geq \overline{M}(x) > 0$. □

We proceed to show that these bounds are in fact the best possible ones. If M were Δ_1^0-computable, then so would be the conditional semimeasure $M(\cdot \mid \cdot)$. Thus we could compute the M-adversarial sequence $z_1 z_2 \ldots$ defined by

$$z_t := \begin{cases} 0 & \text{if } M(1 \mid z_{<t}) > \frac{1}{2}, \\ 1 & \text{otherwise.} \end{cases}$$

The sequence $z_1 z_2 \ldots$ corresponds to a computable deterministic measure μ. However, we have $M(z_{1:t}) \leq 2^{-t}$ by construction, so dominance $M(x) \geq w_\mu \mu(x)$ with $w_\mu > 0$ yields a contradiction with $t \to \infty$:

$$2^{-t} \geq M(z_{1:t}) \geq w_\mu \mu(z_{1:t}) = w_\mu > 0$$

By the same argument, the normalized Solomonoff prior M_{norm} cannot be Δ_1^0-computable. However, since it is a measure, Σ_1^0- or Π_1^0-computability would entail Δ_1^0-computability.

For $\overline{M}$ and $\overline{M}_{\text{norm}}$ we prove the following two lower bounds for specific universal Turing machines.

Theorem 4 ($\overline{M}$ is not Limit Computable). *There is a universal Turing machine U' such that the set $\{(x, q) \mid \overline{M}_{U'}(x) > q\}$ is not in Δ_2^0.*

Proof. Assume the contrary, let A be Π_2^0 but not Δ_2^0, and let S be a computable relation such that

$$n \in A \iff \forall k \exists i \; S(n, k, i). \tag{5}$$

For each $n \in \mathbb{N}$, we define the program p_n as follows.

```
output 1^{n+1}0
k := 0
while true:
    i := 0
    while not S(n,k,i):
        i := i + 1
    k := k + 1
    output 0
```

Each program p_n always outputs $1^{n+1}0$. Furthermore, the program p_n outputs the infinite string $1^{n+1}0^\infty$ if and only if $n \in A$ by (5). We define U' as follows using our reference machine U.

- $U'(1^{n+1}0)$: Run p_n.
- $U'(00p)$: Run $U(p)$.
- $U'(01p)$: Run $U(p)$ and bitwise invert its output.

By construction, U' is a universal Turing machine. No p_n outputs a string starting with $0^{n+1}1$, therefore $\overline{M}_{U'}(0^{n+1}1) = \frac{1}{4}\big(\overline{M}_U(0^{n+1}1) + \overline{M}_U(1^{n+1}0)\big)$. Hence

$$\begin{aligned}\overline{M}_{U'}(1^{n+1}0) &= 2^{-n-2}\mathbb{1}_A(n) + \tfrac{1}{4}\overline{M}_U(1^{n+1}0) + \tfrac{1}{4}\overline{M}_U(0^{n+1}1)\\ &= 2^{-n-2}\mathbb{1}_A(n) + \overline{M}_{U'}(0^{n+1}1)\end{aligned}$$

If $n \notin A$, then $\overline{M}_{U'}(1^{n+1}0) = \overline{M}_{U'}(0^{n+1}1)$. Otherwise, we have $|\overline{M}_{U'}(1^{n+1}0) - \overline{M}_{U'}(0^{n+1}1)| = 2^{-n-2}$.

Now we assume that $\overline{M}_{U'}$ is limit computable, i.e., there is a computable function $\phi : \mathcal{X}^* \times \mathbb{N} \to \mathbb{Q}$ such that $\lim_{k\to\infty} \phi(x,k) = \overline{M}_{U'}(x)$. We get that

$$n \in A \iff \lim_{k\to\infty} \phi(0^{n+1}1, k) - \phi(1^{n+1}0, k) > 2^{-n-3},$$

thus A is limit computable, a contradiction. □

Corollary 5 ($\overline{M}_{\text{norm}}$ is not Σ_2^0- or Π_2^0-computable). *There is a universal Turing machine U' such that $\{(x,q) \mid \overline{M}_{\text{norm}U'}(x) > q\}$ is not in Σ_2^0 or Π_2^0.*

Proof. Since $\overline{M}_{\text{norm}} = c \cdot \overline{M}$, there exists a $k \in \mathbb{N}$ such that $2^{-k} < c$ (even if we do not know the value of k). We can show that the set $\{(x,q) \mid \overline{M}_{\text{norm}U'}(x) > q\}$ is not in Δ_2^0 analogously to the proof of Theorem 4, using

$$n \in A \iff \lim_{k\to\infty} \phi(0^{n+1}1, k) - \phi(1^{n+1}0, k) > 2^{-k-n-3}.$$

If $\overline{M}_{\text{norm}}$ were Σ_2^0-computable or Π_2^0-computable, this would imply that $\overline{M}_{\text{norm}}$ is Δ_2^0-computable since $\overline{M}_{\text{norm}}$ is a measure, a contradiction. □

Since $M(\epsilon) = 1$, we have $M(x \mid \epsilon) = M(x)$, so the conditional probability $M(xy \mid x)$ has at least the same complexity as M. Analogously for M_{norm} and $\overline{M}_{\text{norm}}$ since they are measures. For $\overline{M}$, we have that $\overline{M}(x \mid \epsilon) = \overline{M}_{\text{norm}}(x)$, so Corollary 5 applies. All that remains to prove is that conditional M is not lower semicomputable.

Theorem 6 (Conditional M is not Lower Semicomputable). *The set $\{(x, xy, q) \mid M(xy \mid x) > q\}$ is not recursively enumerable.*

Proof. Assume to the contrary that $M(xy \mid x)$ is lower semicomputable. According to [8, Thm. 12] there is an infinite string $z_{1:\infty}$ such that $z_{2t} = z_{2t-1}$ for all $t > 0$ and

$$\liminf_{t\to\infty} M(z_{1:2t} \mid z_{<2t}) < 1. \tag{6}$$

Define the semimeasure

$$\nu(x_{1:t}) := \begin{cases} \prod_{k=1}^{\lceil t/2 \rceil} M(x_{<2k} \mid x_{<2k-1}) & \text{if } \forall 0 < 2k \leq t \; x_{2k} = x_{2k-1} \\ 0 & \text{otherwise.} \end{cases}$$

Since we assume $M(x_{<2k} \mid x_{<2k-1})$ to be lower semicomputable, ν is lower semicomputable. Therefore there is a constant $c > 0$ such that $M(x) \geq c\nu(x)$ for all $x \in \mathcal{X}^*$. With the chain rule we get for even-lengthed x with $x_{2k} = x_{2k-1}$

$$c \leq \frac{M(x)}{\nu(x)} = \frac{\prod_{i=1}^{t} M(x_{1:i} \mid x_{<i})}{\prod_{k=1}^{t/2} M(x_{<2k} \mid x_{<2k-1})} = \prod_{k=1}^{t/2} M(x_{1:2k} \mid x_{<2k}).$$

Plugging in the sequence $z_{1:\infty}$, we get a contradiction with (6):

$$0 < c \leq \prod_{k=1}^{t} M(z_{1:2k} \mid z_{<2k}) \xrightarrow{t \to \infty} 0 \qquad \square$$

4 The Complexity of Knowledge-Seeking

In general reinforcement learning the 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 $(\mathcal{A} \times \mathcal{E})^*$. We use $æ \in \mathcal{A} \times \mathcal{E}$ to denote one interaction cycle, and $æ_{1:t}$ to denote a history of length t. A *policy* is a function $\pi : (\mathcal{A} \times \mathcal{E})^* \to \mathcal{A}$ mapping each history to the action taken after seeing this history. We assume $\mathcal{A}$ and $\mathcal{E}$ to be finite.

The environment can be stochastic, but is assumed to be semicomputable. In accordance with the AIXI literature [5], we model environments as lower semicomputable *chronological conditional semimeasures* (LSCCCSs). The class of of all LSCCCSs is denoted with $\mathcal{M}$. A *conditional semimeasure* ν takes a sequence of actions $a_{1:t}$ as input and returns a semimeasure $\nu(\cdot \parallel a_{1:t})$ over $\mathcal{E}^\sharp$. A conditional semimeasure ν is *chronological* iff percepts at time t do not depend on future actions, i.e., $\nu(e_{1:t} \parallel a_{1:k}) = \nu(e_{1:t} \parallel a_{1:t})$ for all $k > t$. Despite their name, conditional semimeasures do *not* specify conditional probabilities; the environment ν is *not* a joint probability distribution on actions and percepts. Here we only care about the computability of the environment ν; for our purposes, chronological conditional semimeasures behave just like semimeasures.

Equivalently to (2), the Solomonoff prior M can be defined as a mixture over all lower semicomputable semimeasures using a lower semicomputable *universal prior* [21]. We generalize this representation to chronological conditional semimeasures: we fix the lower semicomputable universal prior $(w_\nu)_{\nu \in \mathcal{M}}$ with $w_\nu > 0$ for all $\nu \in \mathcal{M}$ and $\sum_{\nu \in \mathcal{M}} w_\nu \leq 1$, given by the reference machine U according to $w_\nu := 2^{-K_U(\nu)}$ [5, Sec. 5.1.2]. The universal prior w gives rise to the *universal mixture* ξ, which is a convex combination of all LSCCCSs $\mathcal{M}$:

$$\xi(e_{<t} \parallel a_{<t}) := \sum_{\nu \in \mathcal{M}} w_\nu \nu(e_{<t} \parallel a_{<t})$$

The universal mixture ξ is analogous to the Solomonoff prior M but defined for reactive environments. Analogously to Theorem 3 (i), the universal mixture ξ is lower semicomputable [5, Sec. 5.10]. Moreover, we have $\xi_{\text{norm}} \geq \xi$, preserving universal dominance analogously to M.

4.1 Knowledge-Seeking Agents

We discuss two variants of knowledge-seeking agents: entropy-seeking agents (Shannon-KSA) [14,16] and information-seeking agents (KL-KSA) [17]. The entropy-seeking agent maximizes the Shannon entropy gain, while the information-seeking agent maximizes the expected Bayesian information gain (KL-divergence) in the universal mixture ξ. These quantities are expressed in the *value function*.

In this section we use a finite lifetime m (possibly dependent on time step t): the knowledge-seeking agent maximizes entropy/information received up to and including time step m. We assume that the function m (of t) is computable.

Definition 7 (Entropy-Seeking Value Function [16, Sec. 6]). *The* entropy-seeking value *of a policy π given history $æ_{<t}$ is*

$$V_H^\pi(æ_{<t}) := \sum_{e_{t:m}} -\xi_{\text{norm}}(e_{1:m} \mid e_{<t} \parallel a_{1:m}) \log_2 \xi_{\text{norm}}(e_{1:m} \mid e_{<t} \parallel a_{1:m})$$

where $a_i := \pi(e_{<i})$ for all $i \geq t$.

Definition 8 (Information-Seeking Value Function [17, Def. 1]). *The* information-seeking value *of a policy π given history $æ_{<t}$ is*

$$V_I^\pi(æ_{<t}) := \sum_{e_{t:m}} \sum_{\nu \in \mathcal{M}} w_\nu \frac{\nu(e_{1:m} \parallel a_{1:m})}{\xi_{\text{norm}}(e_{<t} \parallel a_{<t})} \log_2 \frac{\nu(e_{1:m} \mid e_{<t} \parallel a_{1:m})}{\xi_{\text{norm}}(e_{1:m} \mid e_{<t} \parallel a_{1:m})}$$

where $a_i := \pi(e_{<i})$ for all $i \geq t$.

We use V^π in places where either of the entropy-seeking or the information-seeking value function can be substituted.

Definition 9 ($(\varepsilon$-$)$Optimal Policy). *The* optimal value function V^* *is defined as $V^*(æ_{<t}) := \sup_\pi V^\pi(æ_{<t})$. A policy π is* optimal *iff $V^\pi(æ_{<t}) = V^*(æ_{<t})$ for all histories $æ_{<t} \in (\mathcal{A} \times \mathcal{E})^*$. A policy π is* ε-optimal *iff $V^*(æ_{<t}) - V^\pi(æ_{<t}) < \varepsilon$ for all histories $æ_{<t} \in (\mathcal{A} \times \mathcal{E})^*$.*

An entropy-seeking agent is defined as an optimal policy for the value function V_H^* and an information-seeking agent is defined as an optimal policy for the value function V_I^*.

The entropy-seeking agent does not work well in stochastic environments because it gets distracted by noise in the environment rather than trying to distinguish environments [17]. Moreover, the unnormalized knowledge-seeking agents may fail to seek knowledge in deterministic semimeasures as the following example demonstrates.

Example 10 (Unnormalized Entropy-Seeking). Suppose we use ξ instead of ξ_{norm} in Definition 7. Fix $\mathcal{A} := \{\alpha, \beta\}$, $\mathcal{E} := \{0, 1\}$, and $m := 1$ (we only care about the entropy of the next percept). We illustrate the problem on a simple class of environments $\{\nu_1, \nu_2\}$:

where transitions are labeled with action/percept/probability. Both ν_1 and ν_2 return a percept deterministically or nothing at all (the environment ends). Only action α distinguishes between the environments. With the prior $w_{\nu_1} := w_{\nu_2} := 1/2$, we get a mixture ξ for the entropy-seeking value function V_H^π. Then $V_H^*(\alpha) \approx 0.432 < 0.5 = V_H^*(\beta)$, hence action β is preferred over α by the entropy-seeking agent. But taking action β yields percept 0 (if any), hence nothing is learned about the environment. ♢

Solomonoff's prior is extremely good at learning: with this prior a Bayesian agent learns the value of its own policy asymptotically (on-policy value convergence) [5, Thm. 5.36]. However, generally it does not learn the result of counterfactual actions that it does not take. Knowledge-seeking agents learn the environment more effectively, because they focus on exploration. Both the entropy-seeking agent and the information-seeking agent are *strongly asymptotically optimal* in the class of all deterministic computable environments [16,17, Thm. 5]: the value of their policy converges to the optimal value in the sense that $V^\pi \to V^*$ almost surely. Moreover, the information-seeking agent also learns to predict the result of counterfactual actions [17, Thm. 7].

4.2 Knowledge-Seeking is Limit Computable

We proceed to show that ε-optimal knowledge-seeking agents are limit computable, and optimal knowledge-seeking agents are in Δ_3^0.

Theorem 11 (Computability of Knowledge-Seeking). *There are limit-computable ε-optimal policies and Δ_3^0-computable optimal policies for entropy-seeking and information-seeking agents.*

Proof. Since ξ, ν, and w_ν are lower semicomputable, the value functions V_H^* and V_I^* are Δ_2^0-computable according to Lemma 2 (iii-v). The claim now follows from the following lemma. □

Lemma 12 (Complexity of (ε-)Optimal Policies [10, Thm. 8 & 11]). *If the optimal value function V^* is Δ_n^0-computable, then there is an optimal policy π^* that is in Δ_{n+1}^0, and there is an ε-optimal policy π^ε that is in Δ_n^0.*

5 A Weakly Asymptotically Optimal Agent in Δ_2^0

In reinforcement learning we are interested in *reward-seeking* policies. Rewards are provided by the environment as part of each percept $e_t = (o_t, r_t)$ where

$o_t \in \mathcal{O}$ is the *observation* and $r_t \in [0,1]$ is the *reward*. In this section we fix a computable discount function $\gamma : \mathbb{N} \to \mathbb{R}$ with $\gamma(t) \geq 0$ and $\sum_{t=1}^{\infty} \gamma(t) < \infty$. The *discount normalization factor* is defined as $\Gamma_t := \sum_{i=t}^{\infty} \gamma(i)$. 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:

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

Definition 13 (Reward-Seeking Value Function [10, Def. 20]). *The* reward-seeking value *of a policy π in environment ν given history $æ_{<t}$ is*

$$V_\nu^\pi(æ_{<t}) := \frac{1}{\Gamma_t} \sum_{m=t}^{\infty} \sum_{e_{t:m}} \gamma(m) r_m \nu(e_{1:m} \mid e_{<t} \parallel a_{1:m})$$

if $\Gamma_t > 0$ and $V_\nu^\pi(æ_{<t}) := 0$ if $\Gamma_t = 0$ where $a_i := \pi(e_{<i})$ for all $i \geq t$.

Definition 14 (Weak Asymptotic Optimality [7, Def. 7]). *A policy π is* weakly asymptotically optimal *in the class of environments $\mathcal{M}$ iff the reward-seeking value converges to the optimal value on-policy in Cesro mean, i.e.,*

$$\frac{1}{t} \sum_{k=1}^{t} \left(V_\nu^*(æ_{<k}) - V_\nu^\pi(æ_{<k}) \right) \xrightarrow{t\to\infty} 0 \quad \nu\text{-almost surely for all } \nu \in \mathcal{M}.$$

Not all discount functions admit weakly asymptotically optimal policies [7, Thm. 8]; a necessary condition is that the effective horizon grows sublinearly [6, Thm. 5.5]. This is satisfied by geometric discounting, but not by harmonic or power discounting [5, Tab. 5.41].

This condition is also sufficient [6, Thm. 5.6]: Lattimore defines a weakly asymptotically optimal agent called *BayesExp* [6, Ch. 5]. BayesExp alternates between phases of exploration and phases of exploitation: if the optimal information-seeking value is larger than ε_t, then BayesExp starts an exploration phase, otherwise it starts an exploitation phase. During an exploration phase, BayesExp follows an optimal information-seeking policy for $H_t(\varepsilon_t)$ steps. During an exploitation phase, BayesExp follows an ξ-optimal reward-seeking policy for one step [6, Alg. 2].

Generally, optimal reward-seeking policies are Π_2^0-hard [10, Thm.16], and for optimal knowledge-seeking policies we only proved that they are Δ_3^0. Therefore we do not know BayesExp to be limit computable, and we expect it not to be. However, we can approximate it using ε-optimal policies preserving weak asymptotic optimality.

Theorem 15 (A Limit-Computable Weakly Asymptotically Optimal Agent). *If there is a nonincreasing computable sequence of positive reals $(\varepsilon_t)_{t\in\mathbb{N}}$ such that $\varepsilon_t \to 0$ and $H_t(\varepsilon_t)/(t\varepsilon_t) \to 0$ as $t \to \infty$, then there is a limit-computable policy that is weakly asymptotically optimal in the class of all computable stochastic environments.*

Proof. Analogously to Theorem 3 (i) we get that ξ is lower semicomputable, and hence the optimal reward-seeking value function V^*_ν is limit computable [10, Lem. 21]. Hence by Lemma 12, there is a limit-computable 2^{-t}-optimal reward-seeking policy π_ξ for the universal mixture ξ [10, Cor. 22]. By Theorem 11 there are limit-computable $\epsilon_t/2$-optimal information-seeking policies π^t_I with lifetime $t + H_t(\varepsilon_t)$. We define a policy π analogously to BayesExp with π^t_I and π_ξ instead of the optimal policies:

> If $V^*_I(æ_{<t}) > \varepsilon_t$ for lifetime $t + H_t(\varepsilon_t)$, then follow π^t_I for $H_t(\varepsilon_t)$ steps.
> Otherwise, follow π_ξ for one step.

Since V^*_I, π_I, and π_ξ are limit computable, the policy π is limit computable. Furthermore, π_ξ is 2^{-t}-optimal and $2^{-t} \to 0$, so $V^{\pi_\xi}_\xi(æ_{<t}) \to V^*_\xi(æ_{<t})$ as $t \to \infty$.

Now we can proceed analogously to the proof of [6, Thm. 5.6], which consists of three parts. First, it is shown that the value of the ξ-optimal reward-seeking policy π^*_ξ converges to the optimal value for exploitation time steps (second branch in the definition of π) in the sense that $V^{\pi^*_\xi}_\mu \to V^*_\mu$. This carries over to the 2^{-t}-optimal policy π_ξ, since the key property is that on exploitation steps, $V^*_I < \varepsilon_t$; i.e., π only exploits if potential knowledge-seeking value is low. In short, we get for exploitation steps

$$V^{\pi_\xi}_\xi(æ_{<t}) \to V^{\pi^*_\xi}_\xi(æ_{<t}) \to V^{\pi^*_\xi}_\mu(æ_{<t}) \to V^*_\mu(æ_{<t}) \text{ as } t \to \infty.$$

Second, it is shown that the density of exploration steps vanishes. This result carries over since the condition $V^*_I(æ_{<t}) > \varepsilon_t$ that determines exploration steps is exactly the same as for BayesExp and π^t_I is $\varepsilon_t/2$-optimal.

Third, the results of part one and two are used to conclude that π is weakly asymptotically optimal. This part carries over to our proof. □

6 Summary

When using Solomonoff's prior for induction, we need to evaluate conditional probabilities. We showed that conditional M and M_{norm} are limit computable (Theorem 3), and that $\overline{M}$ and $\overline{M}_{\text{norm}}$ are not limit computable (Theorem 4 and Corollary 5); see Table 1 on page 365. This result implies that we can approximate M or M_{norm} for prediction, but not the measure mixture $\overline{M}$ or $\overline{M}_{\text{norm}}$.

In some cases, normalized priors have advantages. As illustrated in Example 10, unnormalized priors can make the entropy-seeking agent mistake the entropy gained from the probability assigned to finite strings for knowledge. From $M_{\text{norm}} \geq M$ we get that M_{norm} predicts just as well as M, and by Theorem 3 we can use M_{norm} without losing limit computability.

Any method that tries to tackle the reinforcement learning problem has to balance between exploration and exploitation. AIXI strikes this balance in the Bayesian way. However, this does not lead to enough exploration [9,15]. Our agent cares more about the present than the future—hence an investment in

form of exploration is discouraged. To counteract this, we can add a knowledge-seeking component to the agent. In Section 4 we discussed two variants of knowledge-seeking agents: entropy-seekers [16] and information-seekers [17]. We showed that ε-optimal knowledge-seeking agents are limit computable and optimal knowledge-seeking agents are Δ_3^0 (Theorem 11).

We set out with the goal of finding a perfect reinforcement learning agent that is limit computable. Weakly asymptotically optimal agents can be considered a suitable candidate, since they are currently the only known general reinforcement learning agents which are optimal in an objective sense [9]. We discussed Lattimore's BayesExp [6, Ch. 5], which relies on Solomonoff induction to learn its environment and on a knowledge-seeking component for extra exploration. Our results culminated in a limit-computable weakly asymptotically optimal agent (Theorem 15). based on Lattimore's BayesExp. In this sense our goal has been achieved.

Acknowledgments. This work was supported by ARC grant DP150104590. We thank Tom Sterkenburg for feedback on the proof of Theorem 6.

References

1. Blackwell, D., Dubins, L.: Merging of opinions with increasing information. The Annals of Mathematical Statistics, 882–886 (1962)
2. Gács, P.: On the relation between descriptional complexity and algorithmic probability. Theoretical Computer Science **22**(1–2), 71–93 (1983)
3. Hutter, M.: A theory of universal artificial intelligence based on algorithmic complexity. Technical Report cs.AI/0004001 (2000). http://arxiv.org/abs/cs.AI/0004001
4. Hutter, M.: New error bounds for Solomonoff prediction. Journal of Computer and System Sciences **62**(4), 653–667 (2001)
5. Hutter, M.: Universal Artificial Intelligence: Sequential Decisions Based on Algorithmic Probability. Springer (2005)
6. Lattimore, T.: Theory of General Reinforcement Learning. PhD thesis, Australian National University (2013)
7. Lattimore, T., Hutter, M.: Asymptotically optimal agents. In: Kivinen, J., Szepesvári, C., Ukkonen, E., Zeugmann, T. (eds.) ALT 2011. LNCS, vol. 6925, pp. 368–382. Springer, Heidelberg (2011)
8. Lattimore, T., Hutter, M., Gavane, V.: Universal prediction of selected bits. In: Kivinen, J., Szepesvári, C., Ukkonen, E., Zeugmann, T. (eds.) ALT 2011. LNCS, vol. 6925, pp. 262–276. Springer, Heidelberg (2011)
9. Leike, J., Hutter, M.: Bad universal priors and notions of optimality. In: Conference on Learning Theory (2015)
10. Leike, J., Hutter, M.: On the computability of AIXI. In: Uncertainty in Artificial Intelligence (2015)
11. Li, M., Vitányi, P.M.B.: An Introduction to Kolmogorov Complexity and Its Applications. Texts in Computer Science, 3rd edn. Springer (2008)
12. Nies, A.: Computability and Randomness. Oxford University Press (2009)
13. Orseau, L.: Optimality issues of universal greedy agents with static priors. In: Hutter, M., Stephan, F., Vovk, V., Zeugmann, T. (eds.) Algorithmic Learning Theory. LNCS, vol. 6331, pp. 345–359. Springer, Heidelberg (2010)

14. Orseau, L.: Universal knowledge-seeking agents. In: Kivinen, J., Szepesvári, C., Ukkonen, E., Zeugmann, T. (eds.) ALT 2011. LNCS, vol. 6925, pp. 353–367. Springer, Heidelberg (2011)
15. Orseau, L.: Asymptotic non-learnability of universal agents with computable horizon functions. Theoretical Computer Science **473**, 149–156 (2013)
16. Orseau, L.: Universal knowledge-seeking agents. Theoretical Computer Science **519**, 127–139 (2014)
17. Orseau, L., Lattimore, T., Hutter, M.: Universal knowledge-seeking agents for stochastic environments. In: Jain, S., Munos, R., Stephan, F., Zeugmann, T. (eds.) ALT 2013. LNCS, vol. 8139, pp. 158–172. Springer, Heidelberg (2013)
18. Rathmanner, S., Hutter, M.: A philosophical treatise of universal induction. Entropy **13**(6), 1076–1136 (2011)
19. Solomonoff, R.: A formal theory of inductive inference. Parts 1 and 2. Information and Control **7**(1), 1–22 and 224–254 (1964)
20. Solomonoff, R.: Complexity-based induction systems: Comparisons and convergence theorems. IEEE Transactions on Information Theory **24**(4), 422–432 (1978)
21. Wood, I., Sunehag, P., Hutter, M.: (Non-)equivalence of universal priors. In: Dowe, D.L. (ed.) Solomonoff Festschrift. LNCS, vol. 7070, pp. 417–425. Springer, Heidelberg (2013)

List of Notation

$:=$	defined to be equal
$\mathbb{N}$	the natural numbers, starting with 0
A, B	sets of natural numbers
$\mathbb{1}_A$	the characteristic function that is 1 if its argument is an element of the set A and 0 otherwise
$\mathcal{X}^*$	the set of all finite strings over the alphabet $\mathcal{X}$
$\mathcal{X}^\infty$	the set of all infinite strings over the alphabet $\mathcal{X}$
$\mathcal{X}^\sharp$	$\mathcal{X}^\sharp := \mathcal{X}^* \cup \mathcal{X}^\infty$, the set of all finite and infinite strings over the alphabet $\mathcal{X}$
x, y	finite or infinite strings, $x, y \in \mathcal{X}^\sharp$
$x \sqsubseteq y$	the string x is a prefix of the string y
ϵ	the empty string, the history of length 0
ε	a small positive real number
$\mathcal{A}$	the (finite) set of possible actions
$\mathcal{O}$	the (finite) set of possible observations
$\mathcal{E}$	the (finite) set of possible percepts, $\mathcal{E} \subset \mathcal{O} \times \mathbb{R}$
M	Solomonoff's prior defined in (2)
$\overline{M}$	the measure mixture defined in (4)
ν_{norm}	Solomonoff normalization of the semimeasure ν defined in (3)
α, β	two different actions, $\alpha, \beta \in \mathcal{A}$
a_t	the action in time step t
e_t	the percept in time step t
o_t	the observation in time step t

r_t	the reward in time step t, bounded between 0 and 1
$æ_{<t}$	the first $t-1$ interactions, $a_1 e_1 a_2 e_2 \dots a_{t-1} e_{t-1}$
γ	the discount function $\gamma : \mathbb{N} \to \mathbb{R}_{\geq 0}$
Γ_t	a discount normalization factor, $\Gamma_t := \sum_{i=t}^{\infty} \gamma(i)$
$H_t(\varepsilon)$	the effective horizon, $H_t(\varepsilon) = \min\{H \mid \Gamma_{t+H}/\Gamma_t \leq \varepsilon\}$
π	a policy, i.e., a function $\pi : (\mathcal{A} \times \mathcal{E})^* \to \mathcal{A}$
V_H^π	the entropy-seeking value of the policy π (see Theorem 7)
V_I^π	the information-seeking value of the policy π (see Theorem 8)
V_ν^π	the reward-seeking value of policy π in environment ν (see Theorem 13)
V^π	the entropy-seeking/information-seeking/reward-seeking value of policy π
V^*	the optimal entropy-seeking/information-seeking/reward-seeking value
ϕ	a computable function
S	a computable relation over natural numbers
n, k, i	natural numbers
t	(current) time step
m	lifetime of the agent (a function of the current time step t)
$\mathcal{M}$	the class of all lower semicomputable chronological conditional semimeasures; our environment class
ν	lower semicomputable semimeasure
μ	computable measure, the true environment
ξ	the universal mixture over all environments in $\mathcal{M}$

Open Questions

1. Can the upper bound of Δ_3^0 for knowledge-seeking policies be improved?
2. Is BayesExp limit computable?
3. Does the lower given in Theorem 4 and Theorem 5 hold for any universal Turing machine?

We expect the answers to questions 1 and 2 to be negative and the answer to question 3 to be positive.