Universal Reinforcement Learning Algorithms: Survey and Experiments

John Aslanides[†], Jan Leike[‡*], Marcus Hutter[†]
[†]Australian National University
[‡]Future of Humanity Institute, University of Oxford
{john.aslanides, marcus.hutter}@anu.edu.au, leike@google.com

Abstract

Many state-of-the-art reinforcement learning (RL) algorithms typically assume that the environment is an ergodic Markov Decision Process (MDP). In contrast, the field of *universal* reinforcement learning (URL) is concerned with algorithms that make as few assumptions as possible about the environment. The universal Bayesian agent AIXI and a family of related URL algorithms have been developed in this setting. While numerous theoretical optimality results have been proven for these agents, there has been no empirical investigation of their behavior to date. We present a short and accessible survey of these URL algorithms under a unified notation and framework, along with results of some experiments that qualitatively illustrate some properties of the resulting policies, and their relative performance on partially-observable gridworld environments. We also present an open-source reference implementation of the algorithms which we hope will facilitate further understanding of, and experimentation with, these ideas.

1 Introduction

The standard approach to reinforcement learning typically assumes that the environment is a fully-observable Markov Decision Process (MDP) [Sutton and Barto, 1998]. Many state-of-the-art applications of reinforcement learning to large state-action spaces are achieved by parametrizing the policy with a large neural network, either directly (e.g. with deep deterministic policy gradients [Silver *et al.*, 2014]) or indirectly (e.g. deep Q-networks [Mnih *et al.*, 2013]). These approaches have yielded superhuman performance on numerous domains including most notably the Atari 2600 video games [Mnih *et al.*, 2015] and the board game Go [Silver *et al.*, 2016]. This performance is due in large part to the scalability of deep neural networks; given sufficient experience and number of layers, coupled with careful optimization, a deep network can learn useful abstract features from high-dimensional input. These algorithms are however restricted in the class of environments that they can plausibly solve, due to the finite capacity of the network architecture and the modelling assumptions that are typically made, e.g. that the optimal policy can be well-approximated by a function of a fully-observable state vector.

In the setting of universal reinforcement learning, we lift the Markov, ergodic, and full-observability assumptions, and attempt to derive algorithms to solve this general class of environments. URL aims to answer the theoretical question: "making as few assumptions as possible about the environment, what constitutes optimal behavior?". To this end several Bayesian, history-based algorithms have been proposed in recent years, central of which is the agent AIXI [Hutter, 2005]. Numerous important open conceptual questions remain [Hutter, 2009], including the need for a relevant, objective, and general optimality criterion [Leike and Hutter, 2015a]. As the field of artifical intelligence research moves inexorably towards AGI, these questions grow in import and relevance.

The contribution of this paper is three-fold: we present a survey of these URL algorithms, and unify their presentation under a consistent vocabulary. We illuminate these agents with an empirical investigation into their behavior and properties. Apart from the MC-AIXI-CTW implementation [Veness *et al.*, 2011] this is the only non-trivial set of experiments relating to AIXI, and is the only set of experiments relating to its variants; hitherto only their asymptotic properties have been studied theoretically. Our third contribution is to present a portable and extensible open-source software framework[1] for experimenting with, and demonstrating, URL algorithms. We also discuss several tricks and approximations that are required to get URL implementations working in practice. Our desire is that this framework will be of use, both for education and research, to the RL and AI safety communities.[2]

2 Literature Survey

This survey covers history-based Bayesian algorithms; we choose history-based algorithms, as these are maximally general, and we restrict ourselves to Bayesian algorithms, as

[*]Now at DeepMind.

[1]The framework is named AIXIJS; the source code can be found at http://github.com/aslanides/aixijs.

[2]A more comprehensive introduction and discussion including more experimental results can be found in the associated thesis at https://arxiv.org/abs/1705.07615.

they are generally both principled and theoretically tractable. The universal Bayesian agent AIXI [Hutter, 2005] is a model of a maximally intelligent agent, and plays a central role in the sub-field of universal reinforcement learning (URL). Recently, AIXI has been shown to be flawed in important ways; in general it doesn't explore enough to be asymptotically optimal [Orseau, 2010], and it can perform poorly, even asymptotically, if given a bad prior [Leike and Hutter, 2015a]. Several variants of AIXI have been proposed to attempt to address these shortfalls: among them are entropy-seeking [Orseau, 2011], information-seeking [Orseau *et al.*, 2013], Bayes with bursts of exploration [Lattimore, 2013], MDL agents [Leike, 2016], Thompson sampling [Leike *et al.*, 2016], and optimism [Sunehag and Hutter, 2015].

It is worth emphasizing that these algorithms are models of rational behavior in general environments, and are not intended to be efficient or practical reinforcement learning algorithms. In this section, we provide a survey of the above algorithms, and of relevant theoretical results in the universal reinforcement learning literature.

2.1 Notation

As we are discussing POMDPs, we distinguish between (hidden) states and percepts, and we take into account histories, i.e. sequences of actions and percepts. States, actions, and percepts use Latin letters, while environments and policies use Greek letters. We use $\mathbb{R}$ as the reals, and $\mathbb{B} = \{0, 1\}$. For sequences over some alphabet $\mathcal{X}$, $\mathcal{X}^k$ is the set of all sequences≈strings of length k over $\mathcal{X}$. We typically use the shorthand $x_{1:k} := x_1 x_2 \dots x_k$ and $x_{<k} := x_{1:k-1}$. Concatenation of two strings x and y is given by xy. We refer to environments and environment models using the symbol ν, and distinguish the *true* environment with μ. The symbol ϵ is used to represent the empty string, while ε is used to represent a small positive number. The symbols $\to$ and $\rightsquigarrow$ are deterministic and stochastic mappings, respectively.

2.2 The General Reinforcement Learning Problem

We begin by formulating the agent-environment interaction. The environment is modelled as a partially observable Markov Decision Process (POMDP). That is, we can assume without loss of generality that there is some hidden state s with respect to which the environment's dynamics are Markovian. Let the state space $\mathcal{S}$ be a compact subset of a finite-dimensional vector space $\mathbb{R}^N$. For simplicity, assume that the action space $\mathcal{A}$ is finite. By analogy with a hidden Markov model, we associate with the environment stochastic dynamics $\mathcal{D} : \mathcal{S} \times \mathcal{A} \rightsquigarrow \mathcal{S}$. Because the environment is in general partially observable, we define a *percept* space $\mathcal{E}$. Percepts are distributed according to a state-conditional percept distribution ν; as we are largely concerned with the agent's perspective, we will usually refer to ν as the environment itself.

The agent selects actions according to a *policy* $\pi(\cdot|\text{æ}_{<t})$, a conditional distribution over $\mathcal{A}$. The agent-environment interaction takes the form of a two-player, turn-based game; the agent samples an action $a_t \in \mathcal{A}$ from its *policy* $\pi(\cdot|\text{æ}_{<t})$, and the environment samples a percept $e_t \in \mathcal{E}$ from $\nu(\cdot|\text{æ}_{<t}a_t)$. Together, they interact to produce a *history*: a sequence of action-percept pairs $h_{<t} \equiv \text{æ}_{<t} := a_1 e_1 \dots a_{t-1} e_{t-1}$. The agent and environment together induce a telescoping distribution over histories, analogous to the *state-visit* distribution in RL:

$$\nu^\pi(\text{æ}_{1:t}) := \prod_{k=1}^{t} \pi(a_k|\text{æ}_{<k})\, \nu(e_k|\text{æ}_{<k}a_k). \tag{1}$$

In RL, percepts consist of (observation, reward) tuples so that $e_t = (o_t, r_t)$. We assume that the reward signal is real-valued, $r_t \in \mathbb{R}$, and make no assumptions about the structure of the $o_t \in \mathcal{O}$. In general, agents will have some utility function u that typically encodes some preferences about states of the world. In the partially observable setting, the agent will have to make inferences from its percepts to world-states. For this reason, the utility function is a function over finite histories of the form $u(\text{æ}_{1:t})$; for agents with an extrinsic reward signal, $u(\text{æ}_{1:t}) = r_t$. The agent's objective is to maximize expected future discounted utility. We assume a general class of convergent *discount functions*, $\gamma_k^t : \mathbb{N} \times \mathbb{N} \to [0, 1]$ with the property $\Gamma_\gamma^t := \sum_{k=t}^{\infty} \gamma_k^t < \infty$. For this purpose, we introduce the *value* function, which in this setting pertains to histories rather than states:

$$V_{\nu\gamma}^{\pi u}(\text{æ}_{<t}) := \mathbb{E}_\nu^\pi \left[\sum_{k=t}^{\infty} \gamma_k^t u(\text{æ}_{1:k}) \,\middle|\, \text{æ}_{<t} \right]. \tag{2}$$

In words, $V_{\nu\gamma}^{\pi u}$ is the expected discounted future sum of reward obtained by an agent following policy π in environment ν under discount function γ and utility function u. For conciseness we will often drop the γ and/or μ subscripts from V when the discount/utility functions are irrelevant or obvious from context; by default we assume geometric discounting and extrinsic rewards. The value of an *optimal policy* is given by the *expectimax* expression

$$\begin{aligned} V_{\nu\gamma}^{\star u}(\text{æ}_{<t}) &= \max_\pi V_{\nu\gamma}^{\pi u} \\ &= \lim_{m\to\infty} \max_{a_t\in\mathcal{A}} \sum_{e_t\in\mathcal{E}} \cdots \max_{a_{t+m}\in\mathcal{A}} \sum_{e_{t+m}\in\mathcal{E}} \Bigg(\\ &\qquad \sum_{k=t}^{t+m} \gamma_k^t u(\text{æ}_{1:k}) \prod_{j=t}^{k} \nu(e_j|\text{æ}_{<j}a_j) \Bigg) \end{aligned} \tag{3}$$

which follows from Eqs. (1) and (2) by jointly maximizing over all future actions and distributing max over $\sum$. The optimal policy is then simply given by $\pi_\nu^\star = \arg\max_\pi V_\nu^\pi$; note that in general the optimal policy may not exist if u is unbounded from above. We now introduce the only non-trivial and non-subjective optimality criterion yet known for general environments [Leike and Hutter, 2015a]: *weak asymptotic optimality*.

Definition 1 (Weak asymptotic optimality; Lattimore & Hutter, 2011)**.** Let the *environment class* $\mathcal{M}$ be a set of environments. A policy π is *weakly asymptotically optimal* in $\mathcal{M}$ if $\forall \mu \in \mathcal{M}$, $V_\mu^\pi \to V_\mu^*$ in mean, i.e.

$$\mu^\pi \left(\lim_{n\to\infty} \frac{1}{n} \sum_{t=1}^{n} \left\{ V_\mu^*(\text{æ}_{<t}) - V_\mu^\pi(\text{æ}_{<t}) \right\} = 0 \right) = 1,$$

where μ^π is the history distribution defined in Equation (1).

AIXI is not in general asymptotically optimal, but both BayesExp and Thompson sampling (introduced below) are. Finally, we introduce the the notion of *effective horizon*, which these algorithms rely on for their optimality.

Definition 2 (ε-Effective horizon; Lattimore & Hutter, 2014)**.** Given a discount function γ, the *ε-effective horizon* is given by

$$H_\gamma^t(\varepsilon) := \min\left\{H \;:\; \frac{\Gamma_\gamma^{t+H}}{\Gamma_\gamma^t} \leq \varepsilon\right\}. \tag{4}$$

In words, H is the horizon that one can truncate one's planning to while still accounting for a fraction equal to $(1-\varepsilon)$ of the realizable return under stationary i.i.d. rewards.

2.3 Algorithms

We consider the class of Bayesian URL agents. The agents maintain a predictive distribution over percepts, that we call a *mixture model* ξ. The agent mixes≈marginalizes over a class of models≈hypotheses≈environments $\mathcal{M}$. We consider countable nonparametric model classes $\mathcal{M}$ so that

$$\xi(e) = \sum_{\nu\in\mathcal{M}} \overbrace{p(e|\nu)}^{\nu(e)} \underbrace{p(\nu)}_{w_\nu}, \tag{5}$$

where we have suppressed the conditioning on history $æ_{<t}a_t$ for clarity. We have identified the agent's credence in hypothesis ν with *weights* w_ν, with $w_\nu > 0$ and $\sum_\nu w_\nu \leq 1$, and we write the probability that ν assigns to percept e as $\nu(e)$. We update with Bayes rule, which amounts to $p(\nu|e) = \frac{p(e|\nu)}{p(e)}p(\nu)$, which induces the sequential weight updating scheme $w_\nu \leftarrow \frac{\nu(e)}{\xi(e)}w_\nu$; see Algorithm 1. We will sometimes use the notation $w_{\nu|æ_{<t}}$ to represent the posterior mass on ν after updating on history $æ_{<t} \in (\mathcal{A}\times\mathcal{E})^*$, and $w(\cdot\,|\,\cdot)$ when referring explicitly to a posterior distribution.

Definition 3 (AIξ; Hutter, 2005)**.** AIξ is the policy that is optimal w.r.t. the Bayes mixture ξ:

$$\begin{aligned}\pi_\xi^\star &:= \arg\max_\pi V_\xi^\pi \\ &= \lim_{m\to\infty} \arg\max_{a_t\in\mathcal{A}} \sum_{e_t\in\mathcal{E}} \cdots \max_{a_m\in\mathcal{A}} \sum_{e_m\in\mathcal{E}} \Bigg(\\ &\qquad \sum_{k=t}^{m} \gamma_k r_k \prod_{j=t}^{k} \sum_{\nu\in\mathcal{M}} w_\nu \nu(e_j|æ_{<j}a_j) \Bigg). \end{aligned} \tag{6}$$

Computational tractability aside, a central issue in Bayesian induction lies in the choice of prior. If computational considerations aren't an issue, we can choose $\mathcal{M}$ to be as broad as possible: the class of all lower semi-computable conditional contextual semimeasures $\mathcal{M}_{\text{CCS}}^{\text{LSC}}$ [Leike and Hutter, 2015b], and using the prior $w_\nu = 2^{-K(\nu)}$, where K is the Kolmogorov complexity. This is equivalent to using Solomonoff's universal prior [Solomonoff, 1978] over strings $M(x) := \sum_{q\,:\,U(q)=x*} 2^{-|q|}$, and yields the AIXI model. AIξ is known to not be asymptotically optimal [Orseau, 2010], and it can be made to perform badly by bad priors [Leike and Hutter, 2015a].

Algorithm 1 Bayesian URL agent [Hutter, 2005]

Inputs: Model class $\mathcal{M} = \{\nu_1, \ldots, \nu_K\}$; prior $w : \mathcal{M} \to (0,1)$; history $æ_{<t}$.

function ACT(π)
  Sample and perform action $a_t \sim \pi(\cdot|æ_{<t})$
  Receive $e_t \sim \nu(\,\cdot\,|æ_{<t}a_t)$
  for $\nu \in \mathcal{M}$ **do**
    $w_\nu \leftarrow \frac{\nu(e_t|æ_{<t}a_t)}{\xi(e_t|æ_{<t}a_t)} w_\nu$
  end for
  $t \leftarrow t+1$
end function

Knowledge-seeking agents (KSA). Exploration is one of the central problems in reinforcement learning [Thrun, 1992], and a principled and comprehensive solution does not yet exist. With few exceptions, the state-of-the-art has not yet moved past ε-greedy exploration [Bellemare *et al.*, 2016; Houthooft *et al.*, 2016; Pathak *et al.*, 2017; Martin *et al.*, 2017]. Intrinsically motivating an agent to explore in environments with sparse reward structure via *knowledge-seeking* is a principled and general approach. This removes the dependence on extrinsic reward signals or utility functions, and collapses the exploration-exploitation trade-off to simply exploration. There are several generic ways in which to formulate a knowledge-seeking utility function for model-based Bayesian agents. We present three, due to Orseau *et al.*:

Definition 4 (Kullback-Leibler KSA; Orseau, 2014)**.** The *KL-KSA* is the Bayesian agent whose utility function is given by the *information gain*

$$\begin{aligned} u_{\text{KL}}(æ_{1:t}) &= \text{IG}(e_t) &(7)\\ &:= \text{Ent}(w(\cdot|æ_{<t})) - \text{Ent}(w(\cdot|æ_{1:t})). &(8)\end{aligned}$$

Informally, the KL-KSA gets rewarded for reducing the entropy (uncertainty) in its model. Now, note that the entropy in the Bayesian mixture ξ can be decomposed into contributions from *uncertainty* in the agent's beliefs w_ν and *noise* in the environment ν. That is, given a mixture ξ and for some percept e such that $0 < \xi(e) < 1$, and suppressing the history $æ_{<t}a_t$,

$$\xi(e) = \sum_{\nu\in\mathcal{M}} \overbrace{w_\nu}^{\text{uncertainty}} \underbrace{\nu(e)}_{\text{noise}}.$$

That is, if $0 < w_\nu < 1$, we say the agent is *uncertain* about whether hypothesis ν is true (assuming there is exactly one $\mu \in \mathcal{M}$ that is the truth). On the other hand, if $0 < \nu(e) < 1$ we say that the environment ν is *noisy* or *stochastic*. If we restrict ourselves to deterministic environments such that $\nu(e) \in \{0,1\}\ \forall\nu\,\forall e$, then $\xi(\cdot) \in (0,1)$ implies that $w_\nu \in (0,1)$ for at least one $\nu \in \mathcal{M}$. This motivates us to define two agents that seek out percepts to which the mixture ξ assigns low probability; in deterministic environments, these will behave like knowledge-seekers.

Definition 5 (Square & Shannon KSA; Orseau, 2011)**.** The *Square* and *Shannon KSA* are the Bayesian agents with utility $u(\text{æ}_{1:t})$ given by $-\xi(\text{æ}_{1:t})$ and $-\log\xi(\text{æ}_{1:t})$ respectively.

Square, Shannon, and KL-KSA are so-named because of the form of the expression when one computes the ξ-expected utility: this is clear for Square and Shannon, and for KL it turns out that the expected information gain is equal to the posterior weighted KL-divergence $\mathbb{E}_\xi[\text{IG}] = \sum_{\nu\in\mathcal{M}} w_{\nu|e}\text{KL}(\nu\|\xi)$ [Lattimore, 2013]. Note that as far as implementation is concerned, these knowledge-seeking agents differ from AIξ *only* in their utility functions.

The following two algorithms (BayesExp and Thompson sampling) are RL agents that add exploration heuristics so as to obtain weak asymptotic optimality, at the cost of introducing an *exploration schedule* $\{\varepsilon_1, \varepsilon_2, \dots\}$ which can be annealed, i.e. $\varepsilon_t \leq \varepsilon_{t-1}$ and $\varepsilon_t \to 0$ as $t \to \infty$.

BayesExp. BayesExp (Algorithm 2) augments the Bayes agent AIξ with bursts of exploration, using the information-seeking policy of KL-KSA. If the expected information gain exceeds a threshold ε_t, the agent will embark on an information-seeking policy $\pi_\xi^{\star,\text{IG}} = \arg\max_\pi V_\xi^{\pi,\text{IG}}$ for one effective horizon, where IG is defined in Equation (7).

Algorithm 2 BayesExp [Lattimore, 2013]

Inputs: Model class $\mathcal{M}$; prior $w : \mathcal{M} \to (0,1)$; exploration schedule $\{\varepsilon_1, \varepsilon_2, \dots\}$.

loop
  if $V_\xi^{*,\text{IG}}(\text{æ}_{<t}) > \varepsilon_t$ **then**
    for $i = 1 \to H_\gamma^t(\varepsilon_t)$ **do**
      ACT $\left(\pi_\xi^{\star,\text{IG}}\right)$
    end for
  else
    ACT $\left(\pi_\xi^\star\right)$
  end if
end loop

Thompson sampling. Thompson sampling (TS) is a very common Bayesian sampling technique, named for [Thompson, 1933]. In the context of general reinforcement learning, it can be used as another attempt at solving the exploration problems of AIξ. From Algorithm 3, we see that TS follows the ρ-optimal policy for an effective horizon before re-sampling from the posterior $\rho \sim w(\cdot|\text{æ}_{<t})$. This commits the agent to a single hypothesis for a significant amount of time, as it samples and tests each hypothesis one at a time.

MDL. The minimum description length (MDL) principle is an inductive bias that implements Occam's razor, originally attributed to [Rissanen, 1978]. The MDL agent greedily picks the simplest probable unfalsified environment σ in its model class and behaves optimally with respect to that environment until it is falsified. If $\mu \in \mathcal{M}$, Algorithm 4 converges with $\sigma \to \mu$. Note that line 2 of Algorithm 4 amounts to maximum likelihood regularized by the Kolmogorov complexity K.

Algorithm 3 Thompson Sampling [Leike *et al.*, 2016]

Inputs: Model class $\mathcal{M}$; prior $w : \mathcal{M} \to (0,1)$; exploration schedule $\{\varepsilon_1, \varepsilon_2, \dots\}$.

1: **loop**
2: Sample $\rho \sim w(\cdot|\text{æ}_{<t})$
3: $d \leftarrow H_t(\epsilon_t)$
4: **for** $i = 1 \to H_t(\varepsilon_t)$ **do**
5: ACT $\left(\pi_\rho^\star\right)$
6: **end for**
7: **end loop**

Algorithm 4 MDL Agent [Lattimore and Hutter, 2011].

Inputs: Model class $\mathcal{M}$; prior $w : \mathcal{M} \to (0,1)$, regularizer constant $\lambda \in \mathbb{R}^+$.

1: **loop**
2: $\sigma \leftarrow \arg\min_{\nu\in\mathcal{M}} \left[K(\nu) - \lambda\sum_{k=1}^t \log\nu(e_k|\text{æ}_{<k}a_k)\right]$
3: ACT $(\pi_\sigma^\star)$
4: **end loop**

3 Implementation

In this section, we describe the environments that we use in our experiments, introduce two Bayesian environment models, and discuss some necessary approximations.

3.1 Gridworld

We run our experiments on a class of partially-observable gridworlds. The gridworld is an $N \times N$ grid of tiles; tiles can be either empty, walls, or stochastic reward dispenser tiles. The action space is given by $\mathcal{A} = \{\leftarrow, \rightarrow, \uparrow, \downarrow, \emptyset\}$, which move the agent in the four cardinal directions or stand still. The observation space is $\mathcal{O} = \mathbb{B}^4$, the set of bit-strings of length four; each bit is 1 if the adjacent tile in the corresponding direction is a wall, and is 0 otherwise. The reward space is small, and integer-valued: the agent receives $r = -1$ for walking over empty tiles, $r = -10$ for bumping into a wall, and $r = 100$ with some fixed probability θ if it is on a reward dispenser tile. There is no observation to distinguish empty tiles from dispensers. In this environment, the optimal policy (assuming unbounded episode length) is to move to the dispenser with highest payout probability θ and remain there, subsequently collecting 100θ reward per cycle in expectation (the environment is non-episodic). In all cases, the agent's starting position is at the top left corner at tile $(0,0)$.

This environment has small action and percept spaces and relatively straightforward dynamics. The challenge lies in coping with perceptual aliasing due to partial observability, dealing with stochasticity in the rewards, and balancing exploration and exploitation. In particular, for a gridworld with n dispensers with $\theta_1 > \theta_2 > \theta_n$, the gridworld presents a non-trivial exploration/exploitation dilemma; see Figure 1.

3.2 Models

Generically, we would like to construct a Bayes mixture over a model class $\mathcal{M}$ that is as rich as possible. Since

we are using a nonparametric model, we are not concerned with choosing our model class so as to arrange for conjugate prior/likelihood pairs. One might consider constructing $\mathcal{M}$ by enumerating all $N \times N$ gridworlds of the class described above, but this is infeasible as the size of such an enumeration explodes combinatorially: using just two tile types we would run out of memory even on a modest 6×6 gridworld since $|\mathcal{M}| = 2^{36} \approx 7 \times 10^{10}$. Instead, we choose a discrete parametrization D that enumerates an interesting subset of these gridworlds. One can think of D as describing a set of parameters about which the agent is uncertain; all other parameters are held constant, and the agent is fully informed of their value. We use this to create the first of our model classes, $\mathcal{M}_{\text{loc}}$. The second, $\mathcal{M}_{\text{Dirichlet}}$, uses a factorized distribution rather than an explicit mixture to avoid this issue.

$\boldsymbol{\mathcal{M}}_{\text{loc}}$. This is a Bayesian mixture parametrized by goal location; the agent knows the layout of the gridworld and knows its dynamics, but is uncertain about the location of the dispensers, and must explore the world to figure out where they are. We construct the model class by enumerating each of these gridworlds. For square gridworlds of side length N, $|\mathcal{M}_{\text{loc}}| = N^2$. From Algorithm 1 the time complexity of our Bayesian updates is $\mathcal{O}(|\mathcal{M}|) = \mathcal{O}(N^2)$.

$\boldsymbol{\mathcal{M}}_{\text{Dirichlet}}$. The Bayes mixture $\mathcal{M}_{\text{loc}}$ is a natural class of gridworlds to consider, but it is quite constrained in that it holds the maze layout and dispenser probabilities fixed. We seek a model that allows the agent to be uncertain about these features. To do this we move away from the mixture formalism so as to construct a bespoke gridworld model.

Let $s_{ij} \in \{\text{EMPTY}, \text{WALL}, \text{DISPENSER}, \dots\}$ be the state of tile (i, j) in the gridworld. The joint distribution over all gridworlds $s_{11}, \dots, s_{NN}$ factorizes across tiles, and the state-conditional percept distributions are Dirichlet over the four tile types. We effectively use a Haldane prior – $\text{Beta}(0, 0)$ – with respect to WALLs and a Laplace prior – $\text{Beta}(1, 1)$ – with respect to DISPENSERs. This model class allows us to make the agent uncertain about the maze layout, including the number, location, and payout probabilities of DISPENSERs. In contrast to $\mathcal{M}_{\text{loc}}$, model updates are $\mathcal{O}(1)$ time complexity, making $\mathcal{M}_{\text{Dirichlet}}$ a far more efficient choice for large N.

This model class incorporates more domain knowledge, and allows the agent to flexibly and cheaply represent a much larger class of gridworlds than using the naive enumeration $\mathcal{M}_{\text{loc}}$. Importantly, $\mathcal{M}_{\text{Dirichlet}}$ is still a Bayesian model – we simply lose the capacity to represent it explicitly as a mixture of the form of Equation (5).

3.3 Agent Approximations

Planning. In contrast to model-free agents, in which the policy is explicitly represented by e.g. a neural network, our model-based agents need to calculate their policy from scratch at each time step by computing the value function in Equation (6). We approximate this infinite forsight with a finite horizon $m > 0$, and we approximate the expectation using Monte Carlo techniques. We implement the ρUCT algorithm [Silver and Veness, 2010; Veness *et al.*, 2011], a history-based version of the UCT algorithm [Kocsis and Szepesvári, 2006]. UCT is itself a variant of Monte Carlo Tree Search (MCTS), a commonly used planning algorithm originally developed for computer Go. The key idea is to try to avoid wasting time considering unpromising sequences of actions; for this reason the action selection procedure within the search tree is motivated by the upper confidence bound (UCB) algorithm [Auer *et al.*, 2002]:

$$a_{\text{UCT}} = \arg\max_{a \in \mathcal{A}} \left(\frac{1}{m(\beta - \alpha)} \hat{V}(\text{æ}_{<t} a) + C \sqrt{\frac{\log T(\text{æ}_{<t})}{T(\text{æ}_{<t} a)}} \right), \tag{9}$$

where $T(\text{æ}_{<t})$ is the number of times the sampler has reached history $\text{æ}_{<t}$, $\hat{V}(\text{æ}_{<t} a)$ is the current estimator of $V(\text{æ}_{<t} a)$, m is the planning horizon, C is a free parameter (usually set to $\sqrt{2}$), and α and β are the minimum and maximum rewards emitted by the environment. In this way, the MCTS planner approximates the expectimax expression in Equation (3), and effectively yields a stochastic policy approximating π defined in Equation (6) [Veness *et al.*, 2011].

In practice, MCTS is very computationally expensive; when planning by forward simulation with a Bayes mixture ξ over $\mathcal{M}$, the worst-case time-complexity of ρUCT is $\mathcal{O}(\kappa m |\mathcal{M}| |\mathcal{A}|)$, where κ is the Monte Carlo sample budget. It is important to note that ρUCT treats the environment model ξ as a black box: it is agnostic to the environment's structure, and so makes no assumptions or optimizations. For this reason, planning in POMDPs with ρUCT is quite wasteful: due to perceptual aliasing, the algorithm considers many plans that are cyclic in the (hidden) state space of the POMDP. This is unavoidable, and means that in practice ρUCT can be very inefficient even in small environments.

Effective horizon. Computing the effective horizon exactly for general discount functions is not possible in general, although approximate effective horizons have been derived for some common choices of γ [Lattimore, 2013]. For most realistic choices of γ and ε, the effective horizon $H_\gamma(\varepsilon)$ is significantly greater than the planning horizon m we can feasibly use due to the prohobitive computational requirements of MCTS [Lamont *et al.*, 2017]. For this reason we use the MCTS planning horizon m as a surrogate for H_γ^t. In practice, all but small values of m are feasible, resulting in short-sighted agents with (depending on the environment and model) suboptimal and highly stochastic policies.

Utility bounds. Recall from Equation (9) that the ρUCT value estimator $\hat{V}(\text{æ}_{1:t})$ is normalized by a factor of $m(\beta - \alpha)$. For reward-based reinforcement learners, α and β are part of the metadata provided to the agent by the environment at the beginning of the agent-environment interaction. For utility-based agents such as KSA, however, rewards are generated intrinsically, and so the agent must calculate for itself what range of utilities it expects to see, so as to correctly normalize its value function for the purposes of planning. For Square and KL-KSA, it is possible to get loose utility bounds $[\alpha, \beta]$ by making some reasonable assumptions. Since $u_{\text{Square}}(e) = -\xi(e)$, we have the bound $-1 \leq u_{\text{Square}} \leq 0$. One can argue that for the vast majority of environments this bound will be tight above, since there will exist percepts e_f that the agent's model has effectively falsified such that $\xi(e_f) \to 0$ as $t \to \infty$.

In the case of the KL-KSA, recall that $u_{\text{KL}}(e) =$

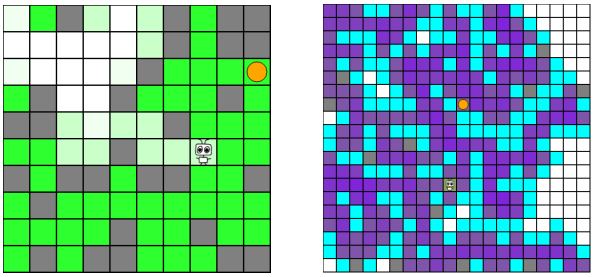

Figure 1: **Left**: An example 10×10 gridworld environment, with the agent's posterior $w_{\nu|æ_{<t}}$ over $\mathcal{M}_{\text{loc}}$ superimposed. Grey tiles are walls, and the agent's posterior is visualized in green; darker shades correspond to tiles with greater probability mass. Here, AIξ has mostly falsified the hypotheses that the goal is in the top-left corner of the maze, and so is motivated to search deeper in the maze, as its model assigns greater mass to unexplored areas. In the ground truth environment μ, the dispenser is located at the tile represented by the orange disc. **Right**: A 20×20 gridworld that is too large to feasibly model with $\mathcal{M}_{\text{loc}}$, with the agent's posterior over $\mathcal{M}_{\text{Dirichlet}}$ superimposed. White tiles are unknown, pale blue tiles are known to be walls, and purple tiles are known to not be walls; darker shades of purple correspond to lower credence that a DISPENSER is present. Notice that despite exploring much of the maze, AIξ has not discovered the profitable reward dispenser located in the upper centre; it has instead settled for a suboptimal dispenser in the lower part of the maze, illustrating the exploration-exploitation tradeoff.

$\text{Ent}(w(\cdot)) - \text{Ent}(w(\cdot|e))$. If we assume a uniform prior $w(\cdot|\epsilon)$, then we have $0 \leq u_{\text{KL}}(e) \leq \text{Ent}(w(\cdot|\epsilon)) \ \forall e \in \mathcal{E}$ since entropy is non-negative over discrete model classes.

Now, in general $u_{\text{Shannon}} = -\log \xi$ is unbounded above as $\xi \to 0$, so unless we can *a priori* place lower bounds on the probability that ξ will assign to an arbitrary percept $e \in \mathcal{E}$, we cannot bound its utility function and therefore cannot calculate the normalization in Equation (9). Therefore, planning with MCTS with the Shannon KSA will be problematic in many environments, as we are forced to make an arbitrary choice of upper bound β.

4 Experiments

We run experiments to investigate and compare the agents' learning performance. Except where otherwise stated, the following experiments were run on a 10×10 gridworld with a single dispenser with $\theta = 0.75$. We average training score over 50 simulations for each agent configuration, and we use $\kappa = 600$ MCTS samples and a planning horizon of $m = 6$. In all cases, discounting is geometric with $\gamma = 0.99$. In all cases, the agents are initialized with a uniform prior.

We plot the training score averaged over $N = 50$ simulation runs, with shaded areas corresponding to one sigma. That is, we plot $\bar{s}_t = \frac{1}{tN} \sum_{i=1}^{N} \sum_{j=1}^{t} s_{ij}$ where s_{ij} is the instantaneous score at time j in run i: this is either reward in the case of reinforcement learners, or fraction of the environment explored in the case of KSA.

Model class. AIξ performs significantly better using the Dirichlet model than with the mixture model. Since the Dirichlet model is less constrained (in other words, less informed), there is more to learn, and the agent is more motivated to explore, and is less susceptible to getting stuck in local maxima. From Figure 2, we see that MC-AIXI-Dirichlet appears to have asymptotically higher variance in its average reward than MC-AIXI. This makes sense since the agent may discover the reward dispenser, but be subsequently incentivized to move away from it and keep exploring, since its model still assigns significant probability to there being better dispensers elsewhere; in contrast, under $\mathcal{M}_{\text{loc}}$, the agent's posterior immediately collapses to the singleton once it discovers a dispenser, and it will greedily stay there ever after. This is borne out by Figure 2, which shows that, on average, AIXI explores significantly more of the gridworld using $\mathcal{M}_{\text{Dirichlet}}$ than with $\mathcal{M}_{\text{loc}}$. These experiments illustrate how AIξ's behavior depends strongly on the model class $\mathcal{M}$.

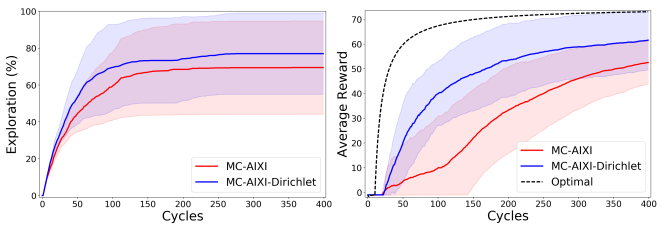

Figure 2: Performance of AIξ is dependent on model class ($\mathcal{M}_{\text{loc}}$ in red, $\mathcal{M}_{\text{Dirichlet}}$ in blue, 'Cycles' $\equiv t$). **Left**: Exploration fraction, $f = 100 \times \frac{N_{\text{visited}}}{N_{\text{reachable}}}$. Note that MC-AIXI and MC-AIXI-Dirichlet both effectively stop exploring quite early on, at around $t \approx 150$. **Right**: Average reward.

KSA. As discussed in Section 2.3, the Shannon- and Square-KSA are entropy-seeking; they are therefore susceptible to environments in which a noise source is constructed so as to trap the agent in a very suboptimal exploratory policy, as the agent gets 'hooked on noise' [Orseau *et al.*, 2013]. The noise source is a tile that emits uniformly random percepts over a sufficiently large alphabet such that the probability of any given percept $\xi(e)$ is lower (and more attractive) than anything else the agent expects to experience by exploring.

KL-KSA explores more effectively than Square- and Shannon-KSA in stochastic environments; see Figure 3. Under the mixture model $\mathcal{M}_{\text{loc}}$, the posterior collapses to a singleton once the agent finds the goal. Given a stochastic dispenser, this becomes the only source of entropy in the environment, and so Square- and Shannon-KSA will remain at the dispenser. In contrast, once the posterior collapses to the minimum entropy configuration, there is no more information to be gained, and so KL-KSA will random walk (breaking ties randomly), and outperform Square and Shannon-KSA marginally in exploration. This difference becomes more marked under the Dirichlet model; although all three agents perform better under $\mathcal{M}_{\text{Dirichlet}}$ due to its being less constrained and having more entropy than $\mathcal{M}_{\text{loc}}$, KL-KSA outperforms the others by a considerable margin; KL-KSA is motivated to explore as it anticipates considerable changes to its model by discovering new areas; see Figure 4.

Exploration. Thompson sampling (TS) is asymptotically optimal in general environments, a property that AIξ lacks. However, in our experiments on gridworlds, TS performs poorly in comparison to AIξ; see Figure 5. This is caused

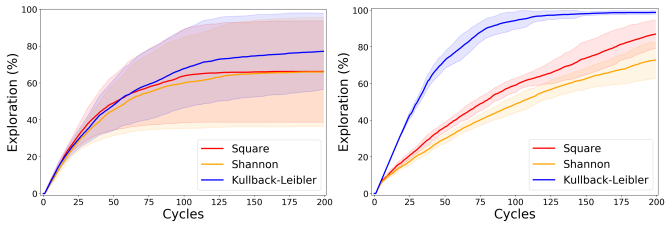

Figure 3: Intrinsic motivation is highly model-dependent, and the information-seeking policy outperforms entropy-seeking in stochastic environments. **Left**: $\mathcal{M}_{\text{loc}}$. **Right**: $\mathcal{M}_{\text{Dirichlet}}$.

by two issues: (1) The parametrization of $\mathcal{M}_{\text{loc}}$ means that the ρ-optimal policy for any $\rho \in \mathcal{M}_{\text{loc}}$ is to seek out some tile (i, j) and wait there for one planning horizon m. For all but very low values of θ or m, this is an inefficient strategy for discovering the location of the dispenser. (2) The performance of TS is strongly constrained by limitations of the planner. If the agent samples an environment ρ which places the dispenser outside its planning horizon – that is, more than m steps away – then the agent will not be sufficiently farsighted to do anything useful. In contrast, AIξ is motivated to continually move around the maze as long as it hasn't yet discovered the dispenser, since ξ will assign progressively higher mass to non-visited tiles as the agent explores more.

Thompson sampling is computationally cheaper than AIξ due to the fact that it plans with only one environment model $\rho \in \mathcal{M}$, rather than with the entire mixture ξ. When given a level-playing field with a time budget rather than a samples budget, the performance gap is reduced; see Figure 5b.

Occam bias. Since each environment $\nu \in \mathcal{M}_{\text{loc}}$ differs by a single parameter and has otherwise identical source code, we cannot use a Kolmogorov complexity surrogate to differentiate them. Instead we opt to order by the environment's index in a row-major order enumeration. On simple deterministic environments (effectively, $\lambda = 0$ in Algorithm 4), the MDL agent significantly outperforms the Bayes agent AIξ with a uniform prior over $\mathcal{M}$; see Figure 6. This is because the MDL agent is biased towards environments with low indices; using the $\mathcal{M}_{\text{loc}}$ model class, this corresponds to environments in which the dispenser is close to the agent's starting position. In comparison, AIξ's uniform prior assigns significant probability mass to the dispenser being deep in the maze. This motivates it to explore deeper in the maze, often neglecting to thoroughly exhaust the simpler hypotheses.

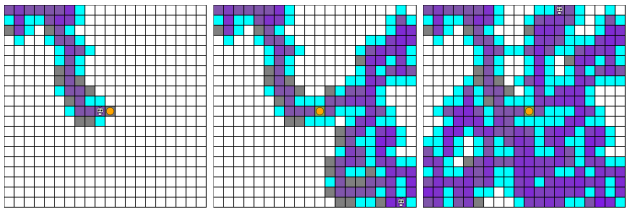

Figure 4: KL-KSA-Dirichlet is highly motivated to explore every reachable tile in the gridworld. **Left** ($t = 36$): The agent begins exploring, and soon stumbles on the dispenser tile. **Center** ($t = 172$): The agent is motivated only by information gain, and so ignores the reward dispenser, opting instead to continue exploring the maze. **Right** ($t = 440$): The agent has now discovered $> 90\%$ of the maze, and continues to gain information from tiles it has already visited as it updates its independent Laplace estimators for each tile.

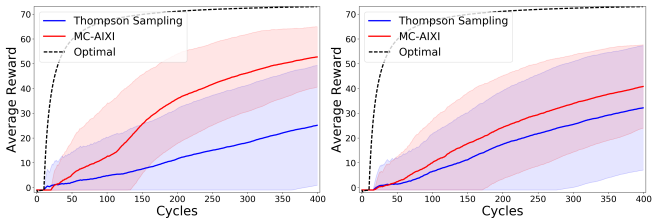

Figure 5: TS typically underperforms AIξ. **Left**: Both agents are given the same MCTS planning parameters: $m = 6$, and a samples budget of $\kappa = 600$. Here, Thompson sampling is unreliable and lacklustre, achieving low mean reward with high variance. **Right**: When each agent is given a horizon of $m = 10$ and a time budget of $t_{\text{max}} = 100$ milliseconds per action, TS performs comparatively better, as the gap to AIξ is narrowed.

5 Conclusion

In this paper we have presented a short survey of state-of-the-art universal reinforcement learning algorithms, and developed experiments that demonstrate the properties of their resulting exploration strategies. We also discuss the dependence of the behavior of Bayesian URL agents on the construction of the model class $\mathcal{M}$, and describe some of the tricks and approximations that are necessary to make Bayesian agents work with ρUCT planning. We have made our implementation open source and available for further development and experimentation, and anticipate that it will be of use to the RL community in the future.

Acknowledgements

We wish to thank Sean Lamont for his assistance in developing the gridworld visualizations used in Figures 1 and 4. We also thank Jarryd Martin and Suraj Narayanan S for proofreading early drafts of the manuscript. This work was supported in part by ARC DP150104590.

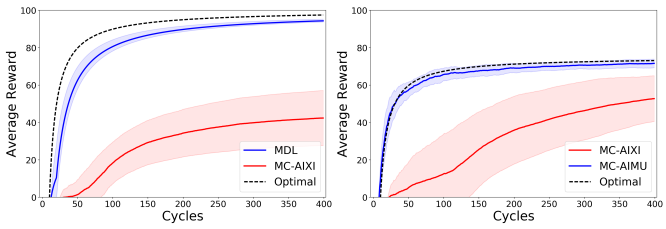

Figure 6: **Left**: MDL vs uniform-prior AIξ in a deterministic ($\theta = 1$) gridworld. **Right**: AIξ compared to the (MC-approximated) optimal policy AIμ with $\theta = 0.75$. Note that MC-AIXI paradoxically performs worse in the deterministic setting than the stochastic one; this is because the posterior over ξ very quickly becomes sparse as hypotheses are rejected, making planning difficult for small m.

References

[Auer *et al.*, 2002] Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002.

[Bellemare *et al.*, 2016] Marc G. Bellemare, Sriram Srinivasan, Georg Ostrovski, et al. Unifying count-based exploration and intrinsic motivation. *CoRR*, abs/1606.01868, 2016.

[Houthooft *et al.*, 2016] Rein Houthooft, Xi Chen, Yan Duan, et al. VIME: Variational information maximizing exploration. In *Advances in Neural Information Processing Systems 29*, pages 1109–1117. 2016.

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

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

[Kocsis and Szepesvári, 2006] Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In *European Conference on Machine Learning*, pages 282–293, 2006.

[Lamont *et al.*, 2017] Sean Lamont, John Aslanides, Jan Leike, and Marcus Hutter. Generalised discount functions applied to a Monte-Carlo AImu implementation. In *Autonomous Agents and Multiagent Systems*, pages 1589–1591, 2017.

[Lattimore, 2013] Tor Lattimore. *Theory of General Reinforcement Learning*. PhD thesis, ANU, 2013.

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

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

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

[Leike, 2016] Jan Leike. *Nonparametric General Reinforcement Learning*. PhD thesis, ANU, 2016.

[Martin *et al.*, 2017] Jarryd Martin, Suraj Narayanan S, Tom Everitt, and Marcus Hutter. Count-based exploration in feature space for reinforcement learning. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, IJCAI'17. AAAI Press, 2017.

[Mnih *et al.*, 2013] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, et al. Playing Atari with deep reinforcement learning. Technical report, Google DeepMind, 2013.

[Mnih *et al.*, 2015] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[Orseau *et al.*, 2013] Laurent Orseau, Tor Lattimore, and Marcus Hutter. Universal knowledge-seeking agents for stochastic environments. In *Algorithmic Learning Theory*, pages 158–172. Springer, 2013.

[Orseau, 2010] Laurent Orseau. Optimality issues of universal greedy agents with static priors. In *Algorithmic Learning Theory*, pages 345–359. Springer, 2010.

[Orseau, 2011] Laurent Orseau. Universal knowledge-seeking agents. In *Algorithmic Learning Theory*, pages 353–367. Springer, 2011.

[Pathak *et al.*, 2017] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell. Curiosity-driven Exploration by Self-supervised Prediction. *ArXiv e-prints*, May 2017.

[Rissanen, 1978] Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978.

[Silver and Veness, 2010] David Silver and Joel Veness. Monte-Carlo planning in large POMDPs. In *Advances in Neural Information Processing Systems 23*, pages 2164–2172. 2010.

[Silver *et al.*, 2014] David Silver, Guy Lever, Nicolas Heess, et al. Deterministic policy gradient algorithms. In Tony Jebara and Eric P. Xing, editors, *International Conference on Machine Learning*, pages 387–395, 2014.

[Silver *et al.*, 2016] David Silver, Aja Huang, Chris J Maddison, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

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

[Sunehag and Hutter, 2015] Peter Sunehag and Marcus Hutter. Rationality, optimism and guarantees in general reinforcement learning. *Journal of Machine Learning Research*, 16:1345–1390, 2015.

[Sutton and Barto, 1998] Richard Sutton and Andrew Barto. *Reinforcement Learning: An Introduction*. MIT, 1998.

[Thompson, 1933] William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, pages 285–294, 1933.

[Thrun, 1992] S. Thrun. The role of exploration in learning control. In *Handbook for Intelligent Control: Neural, Fuzzy and Adaptive Approaches*. 1992.

[Veness *et al.*, 2011] Joel Veness, Kee Siong Ng, Marcus Hutter, William Uther, and David Silver. A Monte-Carlo AIXI approximation. *Journal of Artificial Intelligence Research*, 40(1):95–142, 2011.