Notation for induction and decision theory

Tsvi Benson-Tilsen

1 Introduction

We can take a universal Garrabrant inductor (UGI), as described here: `https://agentfoundations.org/item?id=941`, to be a canonical model for the reasoning procedure of a resource-bounded agent making decisions under logical uncertainty. Here I'll fix some notation for working with UGIs and defining agents using UGIs as world models. This is unpolished and is meant only as a reference.

2 Garrabrant inductors

Table 1: Basic concepts, universal Garrabrant inductors

terminology	notation	type	definition
unit interval	$\mathbb{I}$	$\mathcal{P}(\mathbb{R})$	$[0,1]$
sequence	$\overline{x}, (x_n)$	$\mathbb{N}^+ \to X$	a sequence $(x_n)_{n:\mathbb{N}^+}$ of elements of X
prefix of a sequence	$\overline{x}_{\leq n}$	X^n	$(x_1, \ldots, x_n)$
bits	2	Space	$\{0,1\}$, discrete topology
bitstring or prefix	σ	$2^{<\omega}$	a finite bitstring, in 2^n for some n
(finite) bitstrings or (finite) prefixes	$2^{<\omega}$	Tree	the infinite binary tree of finite bitstrings
infinite bitstring or bit sequence	$\overline{b}$	2^ω	a sequence (b_n) with $b_n : 2$
sequence space or Cantor space	2^ω	Space	the space of infinite bitstrings with the product topology
distributions over $\mathcal{S}$	$\Delta(\mathcal{S})$	Space	the space of probability measures on the space $\mathcal{S}$
belief state	$\mathbb{P}$	$\Delta(2^\omega)$	a distribution over 2^ω specified by finitely many values $\mathbb{P}(\sigma) : \mathbb{Q}$
universal Garrabrant inductor (UGI)	$\overline{\mathbb{P}}, (\mathbb{P}_n)$	$\mathbb{N}^+ \to \Delta(2^\omega)$	a sequence of belief states satisfying the Garrabrant induction criterion
limiting beliefs	$\mathbb{P}_\infty$	$\Delta(2^\omega)$	$\mathbb{P}_\infty(\sigma) := \lim_{n:\omega} \mathbb{P}_n(\sigma)$
logical Garrabrant inductor (LGI)	$\overline{\mathbb{P}^T}, (\mathbb{P}^T_n)$	$\mathbb{N}^+ \to \Delta(2^\Lambda)$	a Garrabrant inductor over T on the sentences Λ, with $\mathbb{P}^T_n(\phi) := \mathbb{P}_n(\phi \vert \bigwedge_{i \leq n} T_i)$

A UGI $\overline{\mathbb{P}}$ is a sequence of probability distributions $\mathbb{P}_n$ over the space 2^ω of infinite bitstrings; the space of such distributions is written as $\Delta(2^\omega)$. To represent these objects in computations, it's convenient to restrict our attention to "belief states", which in this context are distributions specified by giving rational values $\mathbb{P}(\sigma)$ of the set of strings extending σ for finitely many $\sigma \in 2^\omega$. The limit beliefs $\mathbb{P}_\infty$ are defined as the limit of the $\mathbb{P}_n$.

We can obtain a Garrabrant inductor over a theory T of our choosing by conditioning a UGI on the running conjunction of the first n sentences of T in some efficient enumeration. Note that "conditioning a UGI on a sentence" requires that we fix an encoding of sentences as indices into infinite bitstrings; this is important to keep in mind when e.g. we want an agent that reasons about results of computations using a UGI. Also note that if we obtain $\mathbb{P}^{\mathsf{PA}}$ and $\mathbb{P}^{\mathsf{ZFC}}$ in this way, there may be a nontrivial relationship between the two, as they are derived from the same underlying universal inductor.

3 Policies and agents

Table 2: Policies, agents

terminology	notation	type	definition
action	a	Act	
actions	Act	Set	often Act $= 2$
observation	o	Obs	
observations	Obs	Set	often Obs $= 2$ or $= \mathbb{N}$ or $= 2^{<\omega}$
policy	π	Π	
policies	Π	Space	$\Pi :=$ Obs $\to$ Act with the product topology if $\lvert\text{Obs}\rvert = \infty$
agent	A	Expr(Π)	
utility function	U	$2^\omega \to \mathbb{R}$	?
agent for a utility function	A^U	Expr(Π)	an agent that values outcomes according to U

An agent is a program that takes observations as inputs and then computes an action. It thereby implements a policy. Often agents have the form

$$A^U(o) := \arg\max_{a \in \text{Act}} \mathbb{E}[U; A^U(o) = a] \ ,$$

where $\mathbb{E}[-; S]$ is the expectation operator for some belief state representing a model of the (possibly) counterfactual world in which S is true. Defining that belief state is the problem of logical counterfactuals. There is a separate problem, which is that it's not clear how to define utility functions so that it is convenient to faithfully express the decision problems we want to express; some possibilities are discussed in the next section. Agents also often have the form

$$A^U(o) := \pi(o) \text{ where } \pi := \arg\max_{\pi' \in \Pi} \mathbb{E}[U; \forall o : \text{Obs}.A^U(o) = \pi'(o)] \ ,$$

in which case we can say that A^U "performs strategy selection with respect to o". If A^U doesn't "take o into account" when selecting its policy, we can also say that A^U "is updateless with respect to o". It seems relatively straightforward to be updateless with respect to empirical observations, but logical observations are less straightforward, as the belief state can learn those facts just by thinking; so care should be taken when defining agents that are supposed to be updateless with respect to logical facts. Sometimes agents select amongst other agents, and then imitate that other agent, as in some recent work on "asymptotic decision theory". Morally, this can be viewed as selecting a policy, where a policy is an entire agent.

4 Some possible notions of expectation and utility function

Below are some possible definitions of a utility function. (The details are unimportant as far as I know; I'm just collecting them here.)

It is tempting to define utility functions as below, but unfortunately it is less straightforward to use such functions to talk about even simple decision problems. The problem seems to me to be a problem of reference. Whereas in the case of (logical) GIs we could simply check the bit that corresponds to the sentence that defines success, here we are trying to define the utility functions "directly on the territory" of bitstrings. It is then unclear how to talk about e.g. results of computations, since e.g. it ought to be impossible to point to "all instantiations" of a computation that we care about as evidenced by bitstrings.

For this reason it may be better to simply use a LGI. This can be done by having the utility function function defined as above, but depending only on e.g. a logical sentence that is true iff the world is good. This can also be done by taking expectations of logically uncertain variables, as in the logical induction paper: `https://intelligence.org/files/LogicalInduction.pdf`.

Another issue is that we'd need utility functions to converge quickly as they are fed more bits; otherwise a UGI will have useless estimates of the counterfactual expected values, even if the UGI has well-developed beliefs about logic. For this reason, it may make more sense to have utility functions be distinguished symbols, which, along with bitstrings, can be bought and sold by traders. The wealth bounds are then given by This allows for preemptive learning, so that even slow utility functions will be estimated well.

One possibility is to have utility functions given as distinguished symbols in the domain of the UGI, separate from the bitstrings. Estimates (perhaps computed as below) are used to compute bounds on the values of traders. Then the estimates should be as good as those made by an LGI.

Table 3: Utility functions, expectations

terminology	notation	type	definition
world valuation (w.v.), utility function (u.f.)	$\mathcal{V}$, $\mathcal{U}$	$2^\omega \to \mathbb{R}$	bounded and measurable by $\mathcal{B}(2^\omega)$ and $\mathcal{B}(\mathbb{R})$
w.v. limit machine, u.f. limit machine	V^l, U^l	$\mathrm{Expr}(2^{<\omega} \to \mathbb{Q})$	a Turing machine where $\lim_{n:\omega} U^l(\overline{b}_{\leq n})$ exists for any $\overline{b} : 2^\omega$
limit computable u.f.	$\mathcal{U}^l, [\![U^l]\!]$	$2^\omega \to \mathbb{R}$	$\mathcal{U}^l(\overline{b}) := \lim_{n:\omega} U^l(\overline{b}_{\leq n})$
u.f. machine	U	$\mathrm{Expr}(2^{<\omega} \to \mathbb{Q})$	$\lim_{n:\omega} U(\overline{b}_{\leq n})$ exists and $\exists$ $\overline{b}$-computable $\overline{q}$ s.t. $\forall n. q_n > \lvert\mathcal{U}(\overline{b}) - U(\overline{b}_{\leq n})\rvert$ and $\lim_{n:\omega} q_n = 0$
computable u.f.	$\mathcal{U}, [\![U]\!]$	$2^\omega \to \mathbb{R}$	$\mathcal{U}(\overline{b}) := \lim_{n:\omega} U(\overline{b}_{\leq n})$
Δ_0 u.f. machine	U^0	$\mathrm{Expr}(2^\omega \to \mathbb{Q})$	a Turing machine that halts given oracle access to any $\overline{b} : 2^\omega$
Δ_0 u.f.	$\mathcal{U}^0, [\![U^0]\!]$	$2^\omega \to \mathbb{R}$	$\mathcal{U}^0(\overline{b}) := U^0(\overline{b})$
expectation of a u.f.	$\mathbb{E}$	$(2^\omega \to \mathbb{R}) \to \mathbb{R}$	$\mathbb{E}(\mathcal{U}) := \int_{\overline{b}:2^\omega} \mathcal{U}(\overline{b}) d\mathbb{P}$
approximation of the expectation of a (limit) computable u.f.	$\overline{\mathbb{E}}, (\mathbb{E}_n)$	$\mathbb{N}^+ \to$ $\mathrm{Expr}(2^{<\omega} \to \mathbb{Q})$ $\to \mathbb{Q}$	for U (U^l) a u.f. (limit) machine, $\mathbb{E}_n(U)$ is $\sum_{\sigma \in 2^n} \mathbb{P}_n(\sigma) U(\sigma)$, and likewise for $\mathbb{E}_n(U^l)$