Reflective Oracles: A Foundation for Game Theory in Artificial Intelligence

Benja Fallenstein and **Jessica Taylor**
Machine Intelligence Research Institute
{benja,jessica}@intelligence.org

Paul F. Christiano
UC Berkeley
paulfchristiano@eecs.berkeley.edu

Abstract

Classical game theory treats players as special: A description of a game contains a full, explicit enumeration of all players. This isn't a realistic assumption for autonomous intelligent agents. Ideally, such agents would be based on a decision-theoretic foundation for game theory in which their coplayers are a non-distinguished part of the agent's environment, but it is nontrivial to find such a foundation. Attempts to model both players and the environment as Turing machines, for example, fail for standard diagonalization reasons.

In this paper, we consider oracle machines with access to a "reflective" oracle, which is able to answer questions about the outputs of other machines with access to the same oracle. These oracles avoid diagonalization by answering some queries randomly. We show that machines with access to a reflective oracle can be used to define rational agents using causal decision theory. These agents model their environment as a probabilistic oracle machine, which may contain other agents as a non-distinguished part.

We show that if such agents interact, they will play a Nash equilibrium, with the randomization in mixed strategies coming from the randomization in the oracle's answers. This can be seen as providing a foundation for classical game theory in which players aren't special.

1 Introduction

Classical decision theory and game theory are founded on the notion of a perfect Bayesian reasoner [1]. Such an agent may be uncertain which of several possible worlds describes the state of its environment, but given any particular possible world, it is able to deduce exactly what outcome each of its available actions will produce [2]. This assumption is, of course, unrealistic [3, 4]: Agents in the real world must necessarily be boundedly rational reasoners, which make decisions with finite computational resources. Nevertheless, the notion of a perfect Bayesian reasoner provides an analytically tractable first approximation to the behavior of real-world agents, and underlies an enormous body of work in artificial intelligence [5], statistics [6], economics [7], and other fields.

In the context of artificial intelligence, Hutter [8] proposes a particularly concrete notion of "possible world": An *environment*, in his sense, is a Turing machine which takes an agent's actions as input and produces a sequence of observations and rewards as output. Given a prior distribution μ over environments, Hutter defines a Bayesian agent AI_μ which acts in such a way as to maximize the expected reward, given μ. As usual, Hutter assumes that for every *particular* environment, AI_μ can compute exactly what observations and rewards a given sequence of actions leads to.

However, AI_μ cannot in general itself be implemented as a Turing machine, which is problematic in game-theoretic contexts (where an agent's environment contains other agents). To see this, consider the game of *Matching Pennies*, in which two players each choose between two actions ("heads" and "tails"); if the players choose the same action, the first player wins a dollar, if they choose differently, the second player wins. Suppose that both players' decision-making processes are Turing machines, and suppose that both players know the exact source code of their environment, including the source code of their opponent's decision-making algorithm.[1] (Certainty about the environment is of course equivalent to a μ which assigns probability 1 to certain environment.) Finally, assume that like AI_μ, both players choose optimally given their information about their environment.

In this set-up, by assumption, both players' decision-making processes are deterministic; each player either definitely plays "heads" or definitely plays "tails". But neither of these possibilities is consistent:

---

[1]The technique of quining (Kleene's second recursion theorem [9]) shows that it is possible to write two programs that have access to each other's source code.

For example, if the first player chooses heads and the second player can predict this, the second player will choose tails, but if the first player can predict this in turn, it will choose tails, contradicting the assumption that it chooses heads.

The problem is caused by the assumption that given its opponent's source code, a player can figure out what action the opponent will choose. One might think that it could simply run its opponent's source code, but if the opponent does the same, both programs will go into an infinite loop. Even giving the players access to a halting oracle does not help, because even though a machine with access to a halting oracle can predict the behavior of an ordinary Turing machine, it cannot in general predict the behavior of another oracle machine.

Classical game theory resolves this problem by allowing players to choose *mixed* strategies (probability distributions over actions); for example, the unique Nash equilibrium of Matching Pennies is for each player to assign "heads" and "tails" probability 0.5 each. However, instead of treating players' decision-making algorithms as computable processes which are an ordinary part of a world with computable laws of physics, classical game theory treats players as special objects. For example, to describe a problem in game-theoretic terms, we must provide an explicit list of all relevant players, even though in the real world, "players" are ordinary physical objects, not fundamentally distinct from objects such as rocks or clouds.

In this paper, we show that it is possible to define a certain kind of *probabilistic* oracle—that is, an oracle which answers some queries non-deterministically— such that a Turing machine with access to this oracle can perform perfect Bayesian reasoning about environments that can themselves be described as oracle machines with access to the same oracle. This makes it possible for players to treat opponents simply as an ordinary part of this environment.

When an environment contains multiple agents playing a game against each other, the probabilistic behavior of the oracle may cause the players' behavior to be probabilistic as well. We show that in this case, the players will always play a Nash equilibrium, and for every particular Nash equilibrium there is an oracle that causes the players to behave according to this equilibrium. In this sense, our work can be seen as providing a foundation for classical game theory, demonstrating that the special treatment of players in the classical theory is not fundamental.

The oracles we consider are not halting oracles; instead, roughly speaking, they allow oracle machines with access to such an oracle to determine the probability distribution of outputs of other machines with access to the same oracle. Because of their ability to deal with self-reference, we refer to these oracles as *reflective oracles*.

2 Reflective Oracles

In many situations, programs would like to predict the output of other programs. They could simulate the other program in order to do this. However, this method fails when there are cycles (e.g. program A is concerned with the output of program B which is concerned with the output of program A). Furthermore, if a procedure to determine the output of another program existed, then it would be possible to construct a liar's paradox of the form "if I return 1, then return 0, otherwise return 1".

These paradoxes can be resolved by using probabilities. Let $\mathcal{M}$ be the set of *probabilistic oracle machines*, defined here as Turing machines which can execute special instructions to (i) flip a coin that has an arbitrary rational probability of coming up heads, and to (ii) call an oracle O, whose behavior might itself be probabilistic.

Roughly speaking, the oracle answers questions of the form: "Is the probability that machine M returns 1 greater than p?" Thus, O takes two inputs, a machine $M \in \mathcal{M}$ and a rational probability $p \in [0, 1] \cap \mathbb{Q}$, and returns either 0 or 1. If M is guaranteed to halt and to output either 0 or 1 itself, we want $O(M, p) = 1$ to mean that the probability that M returns 1 (when run with O) is at least p, and $O(M, p) = 0$ to mean that it is at most p; if it is equal to p, both conditions are true, and the oracle may answer randomly. In summary,

$$\begin{aligned}
\mathbb{P}(M^O() = 1) \;>\; p &\implies \mathbb{P}(O(M, p) = 1) \;=\; 1 \\
\mathbb{P}(M^O() = 1) \;<\; p &\implies \mathbb{P}(O(M, p) = 0) \;=\; 1
\end{aligned} \quad (1)$$

where we write $\mathbb{P}(M^O() = 1)$ for the probability that M returns 1 when run with oracle O, and $\mathbb{P}(O(M, p) = 1)$ for the probability that the oracle returns 1 on input (M, p). We assume that different calls to the oracle are stochastically independent events (even if they are about the same pair (M, p)); hence, the behavior of an oracle O is fully specified by the probabilities $\mathbb{P}(O(M, p) = 1)$.

Definition A *query* (with respect to a particular oracle O) is a pair (M, p), where $p \in [0, 1] \cap \mathbb{Q}$ and $M^O()$ is a probabilistic oracle machine which almost surely halts and returns an element of $\{0, 1\}$.

Definition An oracle is called *reflective on R*, where R is a set of queries, if it satisfies the two conditions displayed above for every $(M, p) \in R$. It is called *reflective* if it is reflective on the set of all queries.

Theorem 2.1. *(i) There is a reflective oracle.*
(ii) For any oracle O and every set of queries R, there is an oracle O' which is reflective on R and satisfies $\mathbb{P}(O'(M, p) = 1) = \mathbb{P}(O(M, p) = 1)$ for all $(M, p) \notin R$.

Proof. For the proof of (ii), see Appendix B; see also Theorem 5.1, which gives a more elementary proof of

a special case. Part (i) follows from part (ii) by choosing R to be the set of all queries and letting O be arbitrary. □

As an example, consider the machine given by $M^O() = 1 - O(M, 0.5)$, which implements a version of the liar paradox by asking the oracle what it will return and then returning the opposite. By the existence theorem, there is an oracle which is reflective on $R = \{(M, 0.5)\}$. This is no contradiction: We can set $\mathbb{P}(O(M, 0.5) = 1) = \mathbb{P}(O(M, 0.5) = 0) = 0.5$, leading the program to output 1 half the time and 0 the other half of the time.

3 From Reflective Oracles to Causal Decision Theory

We now show how reflective oracles can be used to implement a perfect Bayesian reasoner. We assume that each possible environment that this agent might find itself in can likewise be modeled as an oracle machine; that is, we assume that the laws of physics are computable by a probabilistic Turing machine with access to the same reflective oracle as the agent. For example, we might imagine our agent as being embedded in a Turing-complete probabilistic cellular automaton, whose laws are specified in terms of the oracle.

We assume that each of the agent's hypotheses about which environments it finds itself in can be modeled by a (possibly probabilistic) "world program" $H^O()$, which simulates this environment and returns a description of what happened. We can then define a machine $W^O()$ which samples a hypothesis H according to the agent's probability distribution and runs $H^O()$. In the sequel, we will talk about $W^O()$ as if it refers to a particular environment, but this machine is assumed to incorporate subjective uncertainty about the laws of physics and the initial state of the world.

We further assume that the agent's decision-making process, $A^O()$, can be modeled as a probabilistic oracle machine embedded in this environment. As a simple example, consider the world program

$$W^O() = \begin{cases} \$20 & \text{if } A^O() = 0 \\ \$15 & \text{otherwise} \end{cases} \qquad (2)$$

In this world, the outcome is \$20 (which in this case means the agent receives \$20) if the agent chooses action 0 and \$15 if the agent chooses action 1.

Our task is to find an appropriate implementation of $A^O()$. Here, we consider agents implementing causal decision theory (CDT) [10], which evaluates actions according to the consequences they cause: For example, if the agent is a robot embedded in a cellular automaton, it might evaluate the expected utility of taking action 0 or 1 by simulating what would happen in the environment if the output signal of its decision-making component were replaced by either 0 or 1.

We will assume that the agent's model of the counterfactual consequences of taking different actions a is described by a machine $W_A^O(a)$, satisfying $W^O() = W_A^O(A^O())$ since in the real world, the agent takes action $a = A^O()$. In our example,

$$W_A^O(a) = \begin{cases} \$20 & \text{if } a = 0 \\ \$15 & \text{otherwise} \end{cases} \qquad (3)$$

We assume that the agent has a utility function over outcomes, $u(\cdot)$, implemented as a lookup table, which takes rational values in $[0, 1]$.[2] Furthermore, we assume that both $W_A^O(0)$ and $W_A^O(1)$ halt almost surely and return a value in the domain of $u(\cdot)$. Causal decision theory then prescribes choosing the action that maximizes expected utility; in other words, we want to find an $A^O()$ such that

$$A^O() = \underset{a}{\text{argmax}} \, \mathbb{E}\left[u\left(W_A^O(a)\right)\right] \qquad (4)$$

In the case of ties, any action maximizing utility is allowed, and it is acceptable for $A^O()$ to randomize.

We cannot compute this expectation by simply running $u(W_A^O(a))$ many times to obtain samples, since the environment might contain other agents of the same type, potentially leading to infinite loops. However, we can find an optimal action by making use of a reflective oracle. This is easiest when the agent has only two actions (0 and 1), but similar analysis extends to any number of actions. Define a machine

$$E^O() := \text{flip}\left(\frac{u(W_A^O(1)) - u(W_A^O(0)) + 1}{2}\right) \qquad (5)$$

where $\text{flip}(p)$ is a probabilistic function that returns 1 with probability p and 0 with probability $1 - p$.

Theorem 3.1. *O is reflective on $\{(E, 1/2)\}$ if and only if $A^O() := O(E, 1/2)$ returns a utility-maximizing action.*

Proof. The demand that $A^O()$ return a utility-maxmizing action is equivalent to

$$\begin{aligned} \mathbb{E}[u(W_A^O(1))] > \mathbb{E}[u(W_A^O(0))] &\implies A^O() = 1 \\ \mathbb{E}[u(W_A^O(1))] < \mathbb{E}[u(W_A^O(0))] &\implies A^O() = 0 \end{aligned} \qquad (6)$$

We have

$$\mathbb{P}(E^O() = 1) = \mathbb{E}\left[\frac{u(W_A^O(1)) - u(W_A^O(0)) + 1}{2}\right] \qquad (7)$$

It is not difficult to check that $\mathbb{E}[u(W_A^O(1))] \gtrless \mathbb{E}[u(W_a^O(0))]$ iff $\mathbb{P}(E^O() = 1) \gtrless 1/2$. Together with

---

[2]Since the meaning of utility functions is invariant under affine transformations, the choice of the particular interval $[0, 1]$ is no restriction.

the definition of $A^O()$, we can use this to rewrite the above conditions as

$$\mathbb{P}(E^O() = 1) > 1/2 \implies O(E, 1/2) = 1$$
$$\mathbb{P}(E^O() = 1) < 1/2 \implies O(E, 1/2) = 0 \qquad (8)$$

But this is precisely the definition of "O is reflective on $\{(E, 1/2)\}$". $\qquad \square$

In order to handle agents which can choose between more than two actions, we can compare action 0 to action 1, then compare action 2 to the best of actions 0 and 1, then compare action 3 to the best of the first three actions, and so on. Adding more actions in this fashion does not substantially change the analysis.

4 From Causal Decision Theory to Nash Equilibria

Since we have taken care to define our agents' world models $W_A^O(a)$ in such a way that they can embed other agents,[3] we need not do anything special to pass from single-agent to multi-agent settings. As in the single-agent case, we model the environment by a program $W^O()$ that contains embedded agent programs $A_1^O, \ldots, A_n^O$ and returns an outcome. We can make the dependency on the agent program explicit by writing $W^O() = F^O(A_1^O(), \ldots, A_n^O())$ for some oracle machine $F^O(\cdots)$. This allows us to define machines $W_i^O(a_i) := F^O(a_i, A_{-i}^O()) := F(A_1^O(), \ldots, A_{i-1}^O(), a_i, A_{i+1}^O(), \ldots, A_n^O())$, representing the causal effects of player i taking action a_i.

We assume that each agent has a utility function $u_i(\cdot)$ of the same type as in the previous subsection. Hence, we can define the agent programs $A_i^O()$ just as before:

$$A_i^O() = O(E_i, 1/2)$$
$$E_i^O() = \text{flip}\left(\frac{u_i(W_i^O(1)) - u_i(W_i^O(0)) + 1}{2}\right)$$

Here, each $E_i^O()$ calls $W_i^O()$, which calls $A_j^O()$ for each $j \neq i$, which refers to the source code of $E_j^O()$, but again, Kleene's second recursion theorem shows that this kind of self-reference poses no theoretical problem [9].

This setup very much resembles the setting of normal-form games. In fact:

Theorem 4.1. *Given an oracle O, consider the n-player normal-form game in which the payoff of player i, given the pure strategy profile $(a_1, \ldots, a_n)$,*

is $\mathbb{E}[u_i(F^O(a_1, \ldots, a_n))]$. *The mixed strategy profile given by $s_i := \mathbb{P}(A_i^O() = 1)$ is a Nash equilibrium of this game if and only if O is reflective on $\{(E_1, 1/2), \ldots, (E_n, 1/2)\}$.*

Proof. For $(s_1, \ldots, s_n)$ to be a Nash equilibrium is equivalent to every player's mixed strategy being a best response; i.e., a pure strategy a_i can only be assigned positive probability if it maximizes

$$\mathbb{E}[u_i(F^O(a_i, A_{-i}^O()))] = \mathbb{E}[u_i(W_i^O(a_i))] \qquad (9)$$

By an application of Theorem 3.1, this is equivalent to O being reflective on $\{(E_i, 1/2)\}$. $\qquad \square$

Note that, in particular, any normal-form game with rational-valued payoffs can be represented in this way by simply choosing F^O to be the identity function. In this case, the theorem shows that every reflective oracle (which exists by Theorem 2.1) gives rise to a Nash equilibrium. In the other direction, Theorem 4.1 together with Theorem 2.1(ii) show that for any Nash equilibrium $(s_1, \ldots, s_n)$ of the normal-form game, there is a reflective oracle such that $\mathbb{P}(A_i^O() = 1) = s_i$.

5 From Nash Equilibria to Reflective Oracles

In the previous section, we showed that a reflective oracle can be used to find Nash equilibria in arbitrary normal-form games. It is interesting to note that we can also go in the other direction: For finite sets R satisfying certain conditions, we can construct normal-form games G_R such that the existence of oracles reflective on R follows from the existence of Nash equilibria in G_R. This existence theorem is a special case of Theorem 2.1, but it not only provides a more elementary proof, but also provides a constructive way of finding such oracles (by applying any algorithm for finding Nash equilibria to G_R).

Definition A set R of queries is *closed* if for every $(M, p) \in R$ and every oracle O, $M^O()$ is guaranteed to only invoke the oracle on pairs $(N, q) \in R$. It is *bounded* if there is some bound $B_R \in \mathbb{N}$ such that for every $(M, p) \in R$ and every oracle O, $M^O()$ is guaranteed to invoke the oracle at most B_R times.

Definition Given a finite set $R = \{(M_1, p_1), \ldots, (M_n, p_n)\}$ and a vector $\vec{x} \in [0, 1]^n$, define $O_{\vec{x}}$ to be the oracle satisfying $\mathbb{P}(O_{\vec{x}}(M_i, p_i) = 1) = x_i$ for $i = 1, \ldots, n$, and $\mathbb{P}(O_{\vec{x}}(M, p) = 1) = 0$ for $(M, p) \notin R$.

Theorem 5.1. *For any finite, closed, bounded set $R = \{(M_1, p_1), \ldots, (M_n, p_n)\}$, there is a normal form game G_R with $m := n \cdot (2B_R + 1)$ players, each of which has two pure strategies, such that for any Nash equilibrium strategy profile $(s_1, \ldots, s_m)$, the oracle $O_{\vec{x}}$ with $\vec{x} := (s_1, \ldots, s_n)$ is reflective on R.*

---

[3]More precisely, we have only required that $W_A^O(a)$ always halt and produce a value in the domain of the utility function $u(\cdot)$. Since all our agents do is to perform a single oracle call, they always halt, making them safe to call from $W_A^O(a)$.

Proof. We divide the $n \cdot (2B_R + 1)$ players in our game into three sets: the *main players* $i = 1, \ldots, n$, the *copy players* $g(i,j) := j \cdot n + i$, and the *auxiliary players* $h(i,j) := (B_R + j) \cdot n + i$, for $i = 1, \ldots, n$, $j = 1, \ldots, B_R$.

The mixed strategy s_i of a main player i will determine the probability that $O_{\vec{x}}(M_i, p_i) = 1$. We will force $s_{g(i,j)} = s_i$, i.e., we will force the mixed strategy of each copy player to equal that of the corresponding main player; thus, the copy players will provide us with independent samples from the Bernoulli(s_i) distribution, allowing us to simulate up to B_R independent calls to $O(M_i, p_i)$. Finally, the auxiliary players are used to enforce the constraint $s_{g(i,j)} = s_i$, by having the copy player $g(i,j)$ play a variant of Matching Pennies against the auxiliary player $h(i,j)$.

In order to define the game's payoff function, note first that by writing out each possible way that the at most B_R oracle calls of $M_i^{O_{\vec{x}}}()$ might come out, we can write the probability that this machine returns 1 as a polynomial,

$$\mathbb{P}(M_i^{O_{\vec{x}}}() = 1) \ = \ \sum_{k=1}^{K} c_{i,k} \prod_{i'=1}^{n} x_{i'}^{d_{i,k,i'}} \qquad (10)$$

where $d_{i,k,i'} \leq B_R$. We want to force the main player i to choose pure strategy 1 if this probability is strictly greater than p_i, pure strategy 0 if it is strictly smaller.

To do so, we set player i's payoff function $u_i(\vec{a})$ to

$$u_i(\vec{a}) \ = \ \begin{cases} \sum_{k=1}^{K} f_{i,k}(\vec{a}), & \text{if } a_i = 1, \\ p_i, & \text{otherwise} \end{cases} \qquad (11)$$

where

$$\begin{aligned} &f_{i,k}(\vec{a}) \\ &= \begin{cases} c_{i,k} & \text{if } a_{g(i',j)} = 1 \ \forall 1 \leq i' \leq n, 1 \leq j \leq d_{i,k,i'} \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$
$$(12)$$

Then, assuming we can guarantee $s_{g(i,j)} = s_i$, the expected payoff of strategy 1 to player i is exactly $\mathbb{P}(M_i^{O_{\vec{x}}}() = 1)$, while the payoff of strategy 0 is always p_i; hence, as desired, the Nash equilibrium conditions force i to choose 1 if the probability is greater than p_i, 0 if it is smaller.

It remains to choose the payoffs $(u_{g(i,j)}(\vec{a}), u_{h(i,j)}(\vec{a}))$ of the copy and auxiliary players. In order to force $s_{g(i,j)} = s_i$, we set these payoffs as follows:

$a_i = 0$		
	$a_{h(i,j)} = 0$	$a_{h(i,j)} = 1$
$a_{g(i,j)} = 0$	$(1,0)$	$(0,0)$
$a_{g(i,j)} = 1$	$(0,1)$	$(1,0)$

(13)

$a_i = 1$		
	$a_{h(i,j)} = 0$	$a_{h(i,j)} = 1$
$a_{g(i,j)} = 0$	$(1,0)$	$(0,1)$
$a_{g(i,j)} = 1$	$(0,0)$	$(1,0)$

(14)

We show in Appendix A that at Nash equilibrium, these payoffs force $s_{g(i,j)} = s_i$. $\qquad\square$

Theorem 5.1 is a special case of Theorem 2.1(i). The proof can be adapted to also show an analog of Theorem 2.1(ii), but we omit the details here.

6 Related Work

Joyce and Gibbard [11] describe one justification for mixed Nash equilibria in terms of causal decision theory. Specifically, they discuss a *self-ratification* condition that extends CDT to cases when one's action is evidence of different underlying conditions that might change which actions are rational. An action self-ratifies if and only if it causally maximizes expected utility in a world model that has been updated on the evidence that this action is taken.

For example, consider the setting of a matching pennies game where players can predict each other accurately. The fact that player A plays "heads" is evidence that player B will predict that player A will play "heads" and play "tails" in response, so player A would then have preferred to play "tails", and so the "heads" action would fail to self-ratify. However, the mixed strategy of flipping the coin *would* self-ratify. Our reflection principle encodes some global constraints on players' mixed strategies that are similar to self-ratification.

The question of how to model agents as an ordinary part of the environment is of interest in the speculative study of human-level and smarter-than-human artificial intelligence [12, 13]. Although such systems are still firmly in the domain of futurism, there has been a recent wave of interest in foundational research aimed at understanding their behavior, in order to ensure that they will behave as intended if and when they are developed [14, 15, 13].

Theoretical models of smarter-than-human intelligence such as Hutter's universally intelligent agent AIXI [8] typically treat the agent as separate from the environment, communicating only through well-defined input and output channels. In the real world, agents run on hardware that is part of the environment, and Orseau and Ring [12] have proposed formalisms for studying *space-time embedded intelligence* running on hardware that is embedded in its environment. Our formalism might be useful for studying idealized models of agents embedded in their environment: While real agents must be boundedly rational, the ability to study perfectly Bayesian space-time embedded intelligence might help to clarify which aspects of realistic

systems are due to bounded rationality, and which are due to the fact that real agents aren't cleanly separated from their environment.

7   Conclusions

In this paper, we have introduced *reflective oracles*, a type of probabilistic oracle which is able to answer questions about the behavior of oracle machines with access to the same oracle. We've shown that such oracle machines can implement a version of causal decision theory, and used this to establish a close relationship between reflective oracles and Nash equilibria.

We have focused on answering queries about oracle machines that halt with probability 1, but the reflection principle presented in Section 2 can be modified to apply to machines that do not necessarily halt. To do so, we replace the condition

$$\mathbb{P}(M^O() = 1) \; < \; p \implies \mathbb{P}(O(M,p) = 0) \; = \; 1 \quad (15)$$

by the condition

$$\mathbb{P}(M^O() \neq 0) \; < \; p \implies \mathbb{P}(O(M,p) = 0) \; = \; 1 \quad (16)$$

This is identical to the former principle if $M^O()$ is guaranteed to halt, but provides sensible information even if there is a chance that $M^O()$ loops. Appendix B proves the existence of reflective oracles satisfying this stronger reflection principle.

The ability to deal with non-halting machines opens up the possibility of applying reflective oracles to simplicity priors such as Solomonoff induction [16]. Solomonoff induction defines a probability distribution over infinite bit sequences by, roughly, choosing a random program and running it. Solomonoff induction deals with computable hypotheses, but is itself uncomputable (albeit computably approximable) because it must deal with the possibility that a randomly chosen program may go into an infinite loop after writing only a finite number of bits on its output tape. A reflective oracle version of Solomonoff induction can deal with a hypothesis space consisting of arbitrary oracle machines, while itself being implementable as an oracle machine; this would make it possible to model a predictor which predicts an environment it is itself embedded in. Similarly, Hutter's universally artificial agent AIXI [8] is a version of the agent AI_μ, discussed in the Introduction, whose distribution over environments is a simplicity prior; a reflective oracle version of AIXI would be able to learn to deal with environments containing other, equally powerful agents. We have explored reflective variants of Solomonoff induction and AIXI in [17].

APPENDIX

A   Nash Equilibria in a Variant of Matching Pennies

Lemma A.1. *Consider an n-player game with three distinguished players, each of which has two pure strategies: Player Row has strategies Up and Down, player Column has strategies Left and Right, and player Matrix has strategies Front and Back. Suppose that the payoffs of (Row, Column) depend only on the strategies of these three players, as follows:*

$(1,0)$	$(0,0)$
$(0,1)$	$(1,0)$

$(1,0)$	$(0,1)$
$(0,0)$	$(1,0)$

where the first matrix indicates the payoffs when Matrix plays Front, and the second matrix indicates the payoffs when Matrix plays Back.

Write p for the probability that Row plays Down, and q for the probability that Matrix plays Back. At Nash equilibrium, we have $p = q$.

Proof. Case 1: $0 < q < 1$. In this case, the game between Row and Column is essentially Matching Pennies, and it is immediate that at Nash equilibrium, Column must mix between strategies. This means that Column must be indifferent: $p(1 - q) = (1 - p)q$. Hence, $p > 0$ (since otherwise $q = 0$), and we can divide the equation by pq, yielding $(1 - q)/q = (1 - p)/p$ or $p = q$.

Case 2: $q \in \{0, 1\}$; w.l.o.g. assume $q = 0$. Suppose that there is a Nash equilibrium with $p > 0$. Then at this equilibrium, Column must play Left; but if Column plays Left, then Row strictly prefers Up, which contradicts $p > 0$. Hence, we must have $p = 0 = q$. $\square$

B   Proof of the Existence Theorem

In this appendix, we prove Theorem 2.1(ii). Thus, suppose that R is a set of queries and O is some oracle; we want to show the existence of an oracle O' which is reflective on R and satisfies $\mathbb{P}(O'(M,p) = 1) = \mathbb{P}(O(M,p) = 1)$ for all $(M,p) \notin R$.

We will describe the behavior of O' by a pair of functions, query : $\mathcal{M} \times ([0,1] \cap \mathbb{Q}) \to [0,1]$ and eval : $\mathcal{M} \to [0,1]$. The first of these gives the distribution of O', i.e., query$(M,p) = \mathbb{P}(O'(M,p) = 1)$. The second gives the distribution of a machine's behavior under O': If M almost surely returns either 0 or 1, then eval$(M) = \mathbb{P}(M^{O'}() = 1)$.

Function pairs (query, eval) can be seen as elements of $A := [0,1]^{\mathcal{M} \times ([0,1] \cap \mathbb{Q})} \times [0,1]^{\mathcal{M}}$, which is a convex and compact subset of the locally convex topological vector space $\mathbb{R}^{\mathcal{M} \times ([0,1] \cap \mathbb{Q})} \times \mathbb{R}^{\mathcal{M}}$ (with the product topology). We now define a correspondence $f : A \to \mathrm{Pow}(A)$, such that fixed points (query, eval) $\in f($query, eval$)$ yield oracles O' of the desired form.

We define f by giving a set of necessary and sufficient conditions for $(\text{query}', \text{eval}') \in f(\text{query}, \text{eval})$. We place three conditions on $\text{query}'(M, p)$: If $(M, p) \in R$ and $\text{eval}(M) > p$, then $\text{query}'(M, p) = 1$; if $(M, p) \in R$ and $\text{eval}(M) < p$, then $\text{query}'(M, p) = 0$; and if $(M, p) \notin R$, then $\text{query}'(M, p) = \mathbb{P}(O(M, p) = 1)$.

To describe the conditions on $\text{eval}'(M)$, we will consider the definition of "probabilistic oracle machine" to include the initial state of the machine's working tapes, so that we can view the state of a machine $M^O()$ after one step of computation as a new machine $N^O()$. Then, any machine M can be classified as performing one of the following operations as its first step of computation: (i) a deterministic computation step, yielding a new state N, in which case $\text{eval}'(M) = \text{eval}(N)$; (ii) a coin flip, yielding a state N with a rational probability p and another state N' with probability $1 - p$, in which case $\text{eval}'(M) = p \cdot \text{eval}(N) + (1 - p) \cdot \text{eval}(N')$; (iii) halting, with the output tape containing 0 (in which case $\text{eval}'(M) = 0$) or 1 (in which case $\text{eval}'(M) = 1$) or some other output (in which case $\text{eval}'(M)$ is arbitrary); or (iv) an invocation of the oracle on a pair (M', p), yielding a new state N if the oracle returns 0 and a different new state N' if it returns 1. In the last case, writing $q := \text{query}(M', p)$, the condition is $\text{eval}'(M) = (1 - q) \cdot \text{eval}(N) + q \cdot \text{eval}(N')$.

Given a fixed point $(\text{query}, \text{eval}) \in f(\text{query}, \text{eval})$, define O' by $\mathbb{P}(O'(M, p) = 1) = \text{query}(M, p)$. Then, it can be shown by induction that for every $T \in \mathbb{N}$ and every $M \in \mathcal{M}$, $\text{eval}(M)$ is $\geq$ the probability that $M^{O'}()$ returns 1 after at most T timesteps, and $\leq$ the probability that it returns something other than 0 within this time bound; in the limit, we obtain

$$\mathbb{P}(M^{O'}() = 1) \ \leq \ \text{eval}(M) \ \leq \ \mathbb{P}(M^{O'}() \neq 0) \quad (17)$$

Together with the conditions on $\text{query}(M, p)$, this shows that

$$\mathbb{P}(M^{O'}() = 1) > p \implies \mathbb{P}(O'(M, p) = 1) = 1$$
$$\mathbb{P}(M^{O'}() = 0) > (1 - p) \implies \mathbb{P}(O'(M, p) = 0) = 1 \quad (18)$$

which is a strengthening of the conditions of Section 2: it is equivalent in the case where $M^{O'}()$ halts with probability 1, but provides information even if $M^{O'}()$ may fail to halt.

It remains to be shown that $f(\cdot)$ has a fixed point. To do so, we employ the infinite-dimensional generalization of Kakutani's fixed-point theorem [18].

It is clear from the definition that $f(\text{query}, \text{eval})$ is non-empty, closed and convex for all $(\text{query}, \text{eval}) \in A$. Hence, to show that f has a fixed point, it is sufficient to show that it has closed graph.

Thus, assume that we have sequences $(\text{query}_n, \text{eval}_n) \rightarrow (\text{query}, \text{eval})$ and $(\text{query}'_n, \text{eval}'_n) \rightarrow (\text{query}', \text{eval}')$, such that $(\text{query}'_n, \text{eval}'_n) \in f(\text{query}_n, \text{eval}_n)$ for every n; we need to show that then, $(\text{query}', \text{eval}') \in f(\text{query}, \text{eval})$.

For the conditions on eval', we can simply take the limit $n \rightarrow \infty$ on both sides of each equation. The condition on $\text{query}'(M, p)$ for $(M, p) \notin R$ is clearly fulfilled, since $\text{query}'_n(M, p)$ is constant in this case. The two remaining conditions on $\text{query}'(M, p)$ are entirely symmetrical; without loss of generality, consider the case $\text{eval}(M) > p$, $(M, p) \in R$.

In this case, since $(\text{query}_n, \text{eval}_n) \rightarrow (\text{query}, \text{eval})$ and convergence is pointwise, there must be an n_0 such that $\text{eval}_n(M) > p$ for all $n \geq n_0$. Since $(\text{query}'_n, \text{eval}'_n) \in f(\text{query}_n, \text{eval}_n)$, it follows that $\text{query}'_n(M, p) = 1$ for all $n \geq n_0$, whence $\text{query}'(M, p) = 1$ as desired. This completes the proof.

References

[1] Roger B. Myerson. *Game Theory. Analysis of Conflict.* Harvard University Press, 1991. ISBN: 9780674341159.

[2] Leonard J. Savage. *The Foundations of Statistics.* Dover books on Mathematics. Dover Publications, 1972. ISBN: 9780486623498.

[3] Gerd Gigerenzer and Reinhard Selten, eds. *Bounded Rationality. The Adaptive Toolbox.* Dahlem Workshop Reports. Cambridge, MA: MIT Press, 2001.

[4] Haim Gaifman. "Reasoning with Limited Resources and Assigning Probabilities to Arithmetical Statements". In: *Synthese* 140.1–2 (2004), pp. 97–119. DOI: 10.1023/B:SYNT.0000029944.99888.a7.

[5] Kevin B. Korb and Ann E. Nicholson. *Bayesian Artificial Intelligence.* Chapman & Hall/CRC Computer Science & Data Analysis. Taylor & Francis, 2003. ISBN: 9780203491294.

[6] George E.P. Box and George C. Tiao. *Bayesian Inference in Statistical Analysis.* Wiley Classics Library. Wiley, 1992. ISBN: 978-0-471-57428-6.

[7] David M. Kreps. *A Course in Microeconomic Theory.* Princeton University Press, 1990. ISBN: 9780691042640.

[8] Marcus Hutter. *Universal Artificial Intelligence. Sequential Decisions Based On Algorithmic Probability.* Texts in Theoretical Computer Science. Berlin: Springer, 2005. DOI: 10.1007/b138233.

[9] Hartley Rogers Jr. *Theory of Recursive Functions and Effective Computability.* New York: McGraw-Hill, 1967.

[10] Paul Weirich. "Causal Decision Theory". In: *The Stanford Encyclopedia of Philosophy.* Ed. by Edward N. Zalta. Winter 2012. 2012.

[11] James Joyce and Allan Gibbard. "Causal Decision Theory". In: *Handbook of Utility Theory. Volume 1: Principles.* Kluwer Academic Publishers, 1998, pp. 627–666.

[12] Laurent Orseau and Mark Ring. "Space-Time Embedded Intelligence". In: *Artificial General Intelligence. 5th International Conference, AGI 2012, Oxford, UK, December 8–11, 2012. Proceedings.* Lecture Notes in Artificial Intelligence 7716. New York: Springer, 2012, pp. 209–218. DOI: 10.1007/978-3-642-35506-6_22.

[13] Nate Soares and Benja Fallenstein. "Aligning Super-intelligence with Human Interests. A Technical Research Agenda". In: *The Technological Singularity: Managing the Journey*. Ed. by Jim Miller et al. Vol. 2. Springer, forthcoming.

[14] Future of Life Institute. *Research Priorities for Robust and Beneficial Artificial Intelligence: an Open Letter*. Future of Life Institute. 2015. URL: `http://futureoflife.org/misc/open_letter` (visited on 08/26/2015).

[15] Nick Bostrom. *Superintelligence. Paths, Dangers, Strategies*. New York: Oxford University Press, 2014.

[16] Ray J. Solomonoff. "A Formal Theory of Inductive Inference. Part I". In: *Information and Control* 7.1 (1964), pp. 1–22. DOI: `10.1016/S0019-9958(64)90223-2`.

[17] Benja Fallenstein, Nate Soares, and Jessica Taylor. "Reflective Variants of Solomonoff Induction and AIXI". In: *Artificial General Intelligence. 8th International Conference, AGI 2015, Berlin, Germany, July 22–25, 2015. Proceedings*. Vol. 9205. Lecture Notes in Artificial Intelligence. Springer International Publishing, 2015, pp. 60–69. ISBN: 978-3-319-21364-4. DOI: `10.1007/978-3-319-21365-1`. URL: `https://intelligence.org/files/ReflectiveSolomonoffAIXI.pdf`.

[18] Ky Fan. "Fixed-Point and Minimax Theorems in Locally Convex Topological Linear Spaces". In: *Proceedings of the National Academy of Sciences of the United States of America* 38.2 (1952), pp. 121–126.