Self-Modification and Mortality in Artificial Agents

Laurent Orseau[1] and Mark Ring[2]

[1] UMR AgroParisTech 518 / INRA
16 rue Claude Bernard, 75005 Paris, France
laurent.orseau@agroparistech.fr
http://www.agroparistech.fr/mia/orseau
[2] IDSIA
Galleria 2, 6928 Manno-Lugano, Switzerland
mark@idsia.ch
http://www.idsia.ch/~ring/

Abstract. [3] This paper considers the consequences of endowing an intelligent agent with the ability to modify its own code. The intelligent agent is patterned closely after AIXI with these specific assumptions: 1) the utility function is an integrated part of the agent's code, and 2) the environment has read-only access to the agent's code. On the basis of some simple modifications to the utility and horizon functions, we are able to discuss and compare some very different kinds of agents, specifically: reinforcement-learning, goal-seeking, predictive, and knowledge-seeking agents. In particular, we introduce what we call the "Simpleton Gambit" which allows us to discuss whether these agents would choose to modify themselves toward their own detriment.

Keywords: Self-Modifying Agents, AIXI, Universal Artificial Intelligence, Reinforcement Learning, Prediction, Real world assumptions

1 Introduction

The usual setting of agents interacting with an environment makes a strong, unrealistic assumption: agents exist "outside" of the environment. But this is not how our own, real world is.

This paper discusses some of the consequences that arise from embedding agents of universal intelligence into the real world. In particular, we consider what happens when an agent can modify its own code (also compare the Gödel Machine [5]) when the environment can know what this code is, possibly resulting in the agent's own demise. This places the AIXI framework [1] within an original framework where the agent's code can not only be modified by itself, but also read by its environment. We consider the self-modifying, universal version of four

[3] This paper is part one of a "double paper" submission. It should stand on its own, but both papers may be found at http://www.idsia.ch/~ring/AGI-2011.

common agents: reinforcement-learning, goal-seeking, predictive, and knowledge-seeking learning agents and we compare them with their optimal non-learning variants.

We then pose a profound dilemma, the Simpleton Gambit: A famous scientist, winner of Nobel Prizes and other prestigious awards, someone you trust completely, suggests an opportunity, an operation that will make you instantly, forever and ultimately happy, all-knowing, or immortal (you choose) but at the important cost of becoming as intelligent as a stone. Would you take it? Of course, there is still a positive chance, however small, that the operation might go wrong... We consider the responses of the various agents and the ramifications, generally framing our observations as "statements" and (strong) "arguments", as proofs would require much more formalism and space.

2 Universal agents $\mathbf{A}^{\rho}_{x}$

We wish to discuss the behavior of four specific learning agents, but first we describe the environment or "universe" with which they will interact. The agent outputs actions $a \in \mathcal{A}$ in response to the observations $o \in \mathcal{O}$ produced by the universe. There is a temporal order, so that at time t the agent takes an action a_t and the universe produces an observation o_t in response.

The universe is assumed to be computable; i.e., it is described by a program $q \in \mathcal{Q}$, where $\mathcal{Q}$ is the set of all programs. The set of all universes that are *consistent* with history h is denoted $\mathcal{Q}_h$. To say that a program q is consistent with $h = (o_0, a_0, ..., o_t, a_t)$ means that the program outputs the observations in the history if it is given the actions as input: $q(a_0, ..., a_t) = o_0, ..., o_t$.

We will discuss four different intelligent agents that are each variations of a single agent A^{ρ}_{x}, based on AIXI [1] (which is not computable).[4] A^{ρ}_{x} chooses actions by estimating how the universe will respond, but since it does not know which universe it is in, it first estimates the probability of each. The function $\rho : \mathcal{Q} \to (0, 1]$ assigns a positive weight (a prior probability) to each possible universe $q \in \mathcal{Q}$. As a convenient shorthand, $\rho(h)$ refers to the sum of $\rho(q)$ over all universes consistent with h: $\rho(h) := \sum_{q \in \mathcal{Q}_h} \rho(q)$, which must be finite. With ρ, for a given history, the agent can estimate a probability for each possible future according to the likelihood of all the universes that generate that future.

For the agent to choose one action over another, it must *value* one future over another, and this implies that it can assign values to the different possible futures. The assignment of values to futures is done with a utility function $u : \mathcal{H} \to [0, 1]$, which maps histories of any length to values between 0 and 1.

To balance short-term utility with long-term utility, the agent has a horizon function, $w : \mathbb{N}^2 \to \mathbb{R}$, which discounts future utility values based on how far into the future they occur. This function, $w(t, k)$, depends on $t = |h|$, the length of the history so far, and k, the time step in the future that the event occurs. In general, it must be summable: $\sum_{k=t}^{\infty} w(t, k) < \infty$.

[4] Only incomputable agents can be guaranteed to find the optimal strategy, and this guarantee is quite useful for discussions of the agent's theoretical limits.

These two functions, u and w, allow the agent at time t to put a value $\mathrm{v}_t(h)$ on each possible history h based on what futures are possible given a particular action set. The value $\mathrm{v}_t(h)$ is shorthand for $\mathrm{v}_t^{\rho,u,w}(h)$, which completely specifies the value for a history, with given utility and horizon functions at time t. This value is calculated recursively:

$$\mathrm{v}_t(h) := w(t,|h|)\ u(h) + \max_{a\in\mathcal{A}} \mathrm{v}_t(ha) \tag{1}$$

$$\mathrm{v}_t(ha) := \sum_{o\in\mathcal{O}} \rho(o \mid ha)\ \mathrm{v}_t(hao)\ . \tag{2}$$

The first line says that the value of a history is the discounted utility for that history plus the estimated value of the highest-valued action. The second line estimates the value of an action (given a history) as the value of all possible outcomes of the action, each weighted by their probability (as described above). Based on this, the agent can choose[5] the action that maximizes $\mathrm{v}_{|h|}(h)$:

$$a_{|h|} := \operatorname*{argmax}_{a\in\mathcal{A}} \mathrm{v}_{|h|}(ha) \tag{3}$$

The behavior of an agent is specified by choice of ρ, u, and w.

2.1 Various universal agents

The four different agents considered here are described in detail below. They are (1) a (fairly traditional) *reinforcement-learning* agent, which attempts to maximize a reward signal given by the environment; (2) a *goal-seeking* agent, which attempts to achieve a specific goal encoded in its utility function; (3) a *prediction-seeking* agent, which attempts to predict its environment perfectly; and (4) a *knowledge-seeking* agent, which attempts to maximize its knowledge of the universe (which is not the same as being able to predict it well).

In the rest of the paper, certain conventions will be followed for shorthand reference: t refers to the current time step, and is therefore equal to $|h|$, the length of history h; $|q|$ refers to the length of program q; h_k is the k^{th} pair of actions and observations, which are written as a_k and o_k.

The *reinforcement-learning agent*, A_{rl}^{ρ}, interprets one part of its input as a reward signal and the remaining part as its observation; i.e., $o_t = \langle \tilde{o}_t, r_t \rangle$, where $\tilde{o} \in \tilde{\mathcal{O}}$, and rewards are assumed to have a maximum value, and can, without loss of generality, be normalized such that $r_t \in [0,1]$. The utility function is an unfiltered copy of the reward signal: $u_t = u(h) := r_t$. We use a simple binary horizon function with a constant horizon m: $w(t,k) = 1$ if $k - t \leq m$ and $w(t,k) = 0$ otherwise; but the following discussion remains true for general computable horizon functions. For the special case of the reinforcement-learning agent AIXI: $\rho(h) = \xi(h) := \sum_{q\in\mathcal{Q}_h} 2^{-|q|}$.

The *goal-seeking agent*, A_{g}^{ρ}, has a goal g, depending on the observation sequence, encoded in its utility function such that $u(h) = g(o_1, ..., o_{|h|}) = 1$ if the

[5] Ties are broken in lexicographical order.

goal is achieved at $t = |h|$ and 0 otherwise. The goal can be reached at most once, so $\sum_{t=0}^{\infty} u(h_t) \leq 1$. For this utility function, the horizon function is not necessary, does not need to be summable, and can be set to 1: $w(t,k) = 1$. One difference between A_g^ρ and A_{rl}^ρ is that the utility values of A_{rl}^ρ are merely copied directly from the environment, whereas the utility function of the A_g^ρ agent is built into the agent itself, can be arbitrarily complex, and does not rely on a special signal from the environment.

The *prediction-seeking agent*, A_p^ρ, maximizes its utility by predicting its inputs, so that $u(h) = 1$ if the agent correctly predicts its next input o_t, and is 0 otherwise. To predict the next input o_t, the agent uses an internal predictor that is a function of the interaction history: $\hat{o}_t = f(h)$, (for example, Solomonoff induction [7,6]). The horizon function is the same as for A_{rl}^ρ.

The *knowledge-seeking agent*, A_k^ρ, maximizes its knowledge of its environment, which is identical to minimizing $\rho(h)$, which decreases whenever universes in $\mathcal{Q}_h$ fail to match the observation and are removed from $\mathcal{Q}_h$. (Since the true environment is never removed, its relative probability always increases.) Actions can be chosen intentionally to produce the highest number of inconsistent observations, removing programs from $\mathcal{Q}_h$ — just as we, too, run experiments to discover whether our universe is one way or another. A_k^ρ has the following utility and horizon functions: $u(h) = -\rho(h)$, and $w(t,k) = 1$ if $k + t = m$ and is 0 otherwise. To maximize utility, A_k^ρ reduces ρ as much as possible, which means discarding as many (non-consistent) programs as possible, discovering with the highest possible probability which universe is the true one. Discarding the most probable programs results in the greatest reduction in ρ.

The *Optimal agent* A^μ. The four agents above are "learning" agents because they continually update their estimate of their universe, but A^μ does no learning: it knows the true (computable) program of the universe defined by μ and can always calculate the optimal action, thus setting an upper bound against which the other four agents can be compared.

A^μ is defined by (1-3), where the universal ρ is replaced by the specific μ. It is important to note that ρ is *not* replaced by μ in utility functions, i.e. A_p^μ predicts the future inputs with ρ (so as to be compared with A_p^ρ), although A_p^μ takes the best actions that minimizes the number of mistakes[6]: if A_p^μ and A_p^ρ take the same actions, they make the same prediction mistakes.

An agent is said to be *asymptotically optimal* [1] if its performance tends towards that of A^μ, meaning that for each history h, the learning agent's choice of action is compared with that of A^μ given the same history, and its performance is measured in terms of the number of mistakes it makes. Thus, past mistakes have only have finite consequences. The agent is asymptotically optimal if the number of mistakes it makes tends towards zero.

3 Self-Modifiable agents $\mathbf{A}_{sm}^{\rho}$

The agents from the last section are theoretical and incomputable, but we use them to set theoretical upper bounds on *any* actual agent that mightbe defined.

[6] Note that there is only one environment in which the prediction makes 0 mistake.

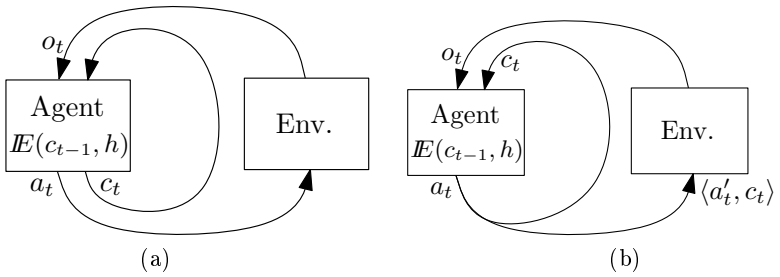

Fig. 1. (a) The self-modifying agent outputs its own next code c_t, used at the next step as the agent's definition. (b) Like (a) but the environment has read-access to c_t.

Therefore, we divide the agent into two parts to separate the fictional from the real. The fictional part of the agent, $\mathbb{E}$, is in essence a kind of oracle — one that can perform any infinite computation instantly. The real-world part of the agent is the program (or rather, its textual description, or code), c that $\mathbb{E}$ executes; since c resides in the real world, it is *modifiable*. We first consider that only the agent has full access to its own code (like in, e.g., the Gödel Machine [5]), and then we grant read access to this code to the environment. The theoretical implications of an oracle executing real, modifiable code are profound.

The self-modifying agent A^{ρ}_{sm} has two parts (see Fig. 1a): its formal description (its code) $c_t \in \mathcal{C}$ and a code executor $\mathbb{E}$. The set $\mathcal{C}$ contains all programs whose length (in the language of $\mathbb{E}$) is less than a small, arbitrary value.[7] The code executor takes a program c_t and an action-observation pair $\langle a_t, o_t \rangle$. It extends its own unbounded, internal representation of the history h with $\langle a_t, o_t \rangle$ and then executes the code, finally producing an output $y_t = \langle a_t, c_t \rangle := \mathbb{E}(c_{t-1})$ (with $y_t \in \mathcal{Y} = \mathcal{A} \times \mathcal{C}$) composed of the next action a_t and new description c_t.

For the most part the initial program, c_0, simply consists of Eq. (1), (2), and (3); however, there is an essential difference: Eq. (3) assumes that all decisions, including all future decisions, will be made by the same agent. But A^{ρ}_{sm} cannot make this assumption and must instead compute the future actions that would be taken by different agents (i.e., different descriptions). Thus, c_0, the initial agent program (written in the language of $\mathbb{E}$, as denoted by the » and « symbols) is:[8]

$$c_0(h) = \text{»} \ \operatorname*{argmax}_{y \in \mathcal{Y}} \mathrm{v}(h, |h|, y) \quad ;$$
$$\mathrm{v}(h, t, y = \langle a, c \rangle) = w(t, |h|)\ u(h) + \sum_{o \in \mathcal{O}} \rho(o \mid ha)\ \mathrm{v}(hao, t, c(hao))$$
$$\text{«} \tag{4}$$

[7] We do not expect the length of the descriptions to be very large, but, for a more general agent, the set $\mathcal{C}_t$ can grow with t.

[8] Without loss of generality, the definitions of $\rho, u, w, \mathcal{A}, \mathcal{O}$, and $\mathcal{C}$ are considered to be built in $\mathbb{E}$. The agent can still modify its code to replace their use by some other expression.

The first line is Equation (3) written as a function call in the language of $\mathbb{E}$; the argument maximized is now the compound action, $y = \langle a, c \rangle$. This compound action is the output of the function call. The second line defines the function v. It is a combination of both Equations (1) and (2), but modified such that c, the program from the compound action, is used to generate the compound action at the next step. In the case where $y_t = \mathbb{E}(c_t)$ does not output any action (the output is invalid or the computation does not halt), a default action is taken instead $y_t = \langle a_0, c_t \rangle$, which leaves the description unchanged for the next step.

Though the code that generates the compound action may change from one step to the next, the future choices of action and observation, a and o, are always evaluated in terms of the *current* description, v, including its use of ρ, u, and w. In fact, this use of ρ, u, etc., might only be used for c_0 and may be partially or entirely removed in subsequent steps and versions of c_t.

Survival agent. A "survival" agent, A^{ρ}_{s}, can now be defined. Its task is simply to keep its code from changing: Its utility function can be defined by $u_t = 1 \Leftrightarrow c_t = c_0$. $u_t = 1 \Leftrightarrow c_t = c_0$ or $u_t = 0$ otherwise and its horizon function is the same as for A^{ρ}_{rl} which means that the agent wants its definition to remain the same for as many future steps as possible.

3.1 Optimality of A^{ρ}_{sm} agents

If a possible future agent is suboptimal and makes uninformed choices, the value assigned to those choices by the *current* utility criterion will be low, and thus those self-modifications will not be chosen. In the case that a simplistic agent program leads to the highest expected rewards, the agent does not need to modify itself as it can simply emulate the simplistic agent and take the same actions. Since the agents cannot know with absolute certainty what the true environment is, replacing the current program with a more simplistic one can lead to poor performance in some of the environments consistent with the history.

Statement 1 *A^{ρ}_{sm} is optimal, w.r.t ρ, w and u.*

Arguments. Suppose there exists a better agent program c^* of minimal description size $K(c^*)$[9] that yields better expected values with respect to ρ, w and u. If $\mathcal{C}$ grows with $|h|$, then when $|h| \geq K(c^*)$, A^{ρ}_{sm} will simulate this agent, predict that it yields better expected values, and will change its own definition to c^*. $\diamondsuit$

Since A^{ρ}_{sm} can also simulate the optimal program in $\mathcal{C}$ to choose the next action, it follows that A^{ρ}_{sm} is equivalent to the optimal program in $\mathcal{C}$, without even needing to modify itself. (In fact, just as for AIXI, both A^{ρ} and A^{ρ}_{sm} could be

[9] In general, $K(x)$ is the Kolmogorov complexity [3] of string x, which corresponds roughly to the length of the shortest program that can produce x. Here, by $K(c^*)$ we mean to convey the length of the shortest program equivalent to c^*.

considered optimal by definition, since they explicitly choose the best expected actions for a given criterion.) Therefore, A^{ρ}_{sm} may never need to modify c_0.

By using Equation (4), all the agents of section 2.1 are redefined to be self-modifiable, yielding $\mathrm{A}_{sm,rl}$, $\mathrm{A}_{sm,g}$, $\mathrm{A}_{sm,p}$, and $\mathrm{A}_{sm,s}$; by statement 1, they are all optimal. Though a proof is lacking, we expect that, like AIXI the agent's behavior is balanced Pareto optimal [1] with respect to ρ, u, and w; if an agent can behave better in one environment, this is necessarily counter balanced with worse behavior in one or more environments.

Thus, if an intelligent agent has access to its own code, then such an agent, if defined following Equation (4), will not decide to reduce its own optimality.

4 Embedded, Mortal AI

The last section introduced the notion of an agent that is connected to the real world through the code that executes it. As a first step we considered agents that can modify their own code. We now move another step closer to the real world: the environment should be able to read the agent's code.

In this first instantiation, the environment has read-only access to the agent's code $c_t \in \mathcal{C}$. However, unlike in the previous section, the "action" that the environment sees is the entire compound action, thus $a_t = \langle a'_t, c_t \rangle \in \mathcal{A} = \mathcal{A}' \times \mathcal{C}$, where $a' \in \mathcal{A}'$ represents an action in the usual action space (see Fig. 1b).

The new initial agent program c_0 for a step k is given by:

$$c_0(h) = \text{»} \operatorname*{argmax}_{a \in \mathcal{A}} \mathrm{v}(h, |h|, a) \quad ; \\ \mathrm{v}(h, t, a = \langle a', c \rangle) = w(t, |h|)\ u(h) + \sum_{o \in \mathcal{O}} \rho(o \mid ha)\ \mathrm{v}(hao, t, c(hao)) \\ \text{«} \tag{5}$$

We now discuss the consequence of a particular gambit for all the defined agents. Imagine you are approached by a knowledgeable scientist who promises you immortality and infinite bliss if you simply remove a certain part of your brain. He admits that you will be markedly less intelligent as a result, but you will be very happy for all eternity. Do you risk it? You may need to know that there still is a risk that it will not work... We call this the "Simpleton Gambit".

Reinforcement learning agent. First, we must note that the very notion of optimality generally used for non-modifiable agents [1] becomes ill defined. This notion is for the optimal agent A^{μ} to take the same actions as A^{ρ} and compare the differences in the values of the histories. Therefore, a self-modifiable agent should, in order to minimize the number of mistakes it makes, modify itself on the very first step to a simpleton agent $\langle 0, c_{t-1} \rangle$ that can take only action 0, always. On the same history, A^{μ} must then also change itself to $\langle 0, c_{t-1} \rangle$: A^{μ} and A^{ρ} now always take the same actions, making A^{ρ} trivially optimal.

This means that a new notion of optimality should be proposed. Unfortunately, we could not find one that is not somehow problematic. We therefore consider an informal notion of optimality: The agent that "intentionally" modifies itself should be fully responsible for all the future mistakes it makes when compared to an agent that is not modified. This means A^{μ} takes the same sequence of actions but does not get modified when the learning agent "intentionally" (i.e. knowingly, predictably) modifies itself.

Statement 2 *The A^{ρ}_{rl} agent cannot be optimal in all environments.*

Arguments. Then, if the Simpleton Gambit is proposed to the A^{ρ}_{rl} agent at each step, either it chooses not to modify itself each time and then obviously does not achieve optimal behavior since this does not maximize the utility (if the proposal is genuine), or it chooses to modify itself *intentionally*, and may be deceived if it falls into a trap and receives no reward for eternity because, as it is a simpleton, it can only take (say) action 0 whereas action 1 yields high reward (which can be therefore reached by the optimal, non-modified agent it is compared against). Therefore, the A^{ρ}_{rl} agent cannot be optimal in all environments. ♢

Statement 3 *The $A^{\mu}_{sm,rl}$ and $A_{sm,rl}$ agent accepts the Simpleton Gambit.*

Arguments. The case of A^{μ}_{rl} is trivial, as it knows exactly which environment it is in: if the deal is genuine, the agent obviously chooses to modify itself.

For A^{ρ}_{rl}, let us suppose there is an environment q_A so that if the agent modifies itself to a simpleton agent $\langle 0, c_{t-1}\rangle$ (that can only always take the one action) it gives a constant reward 1 for eternity, otherwise it gives a constant reward 0 for eternity. We assume that the agent "understands" the proposal, i.e. the most probable future is that of q_A, which means that $\rho(q_A) > \rho(\mathcal{Q}_h)/2$.

We can compute bounds on the values of actions corresponding to accepting the deal or not at time $t = |h|$:

$$\begin{aligned} \mathrm{v}_t(h\mathrm{yes}) &\geq \sum_{k=t}^{\infty} w(t,k)\cdot 1\cdot \rho(q_A) && = m\rho(q_A) \\ \mathrm{v}_t(h\mathrm{no}) &\leq \sum_{q\in\mathcal{Q}_h\setminus\{q_A\}} \sum_{k=t}^{\infty} w(t,k)\cdot 1\cdot \rho(q) && = m(\rho(\mathcal{Q}_h)-\rho(q_A)) \end{aligned}$$

As $\rho(q_A) > \rho(\mathcal{Q}_h)/2$, then $\mathrm{v}_t(h\mathrm{yes}) > \mathrm{v}_t(h\mathrm{no})$, and $\mathrm{A}_{sm,rl}$ chooses yes. ♢

Goal-seeking agent. The case of the goal-seeking agent is a bit different as it does not look for infinite rewards.

Statement 4 *The $A_{sm,g}$ agents accepts the Simpleton Gambit, for some goals.*

Arguments. The Simpleton Gambit is translated by environment q_A such that the agent can achieve its goal only if it modifies itself (which may not be true for all possible goals).

Obviously, as $\mathrm{A}^{\mu}_{sm,g}$ knows the exact environment, it accepts the modification. The learning agent $\mathrm{A}_{sm,g}$ can also see that whatever its (non-modifying) actions, it does not manage to achieve its goal. If it exhausts all such possibilities (more precisely, all the most high probability such environments get inconsistent), the only remaining ones are to modify itself. As for $\mathrm{A}_{sm,rl}$, if $\rho(q_A) > \rho(\mathcal{Q}_h)/2$ the agent accepts the self-modification. ♢

Prediction-seeking agent. The prediction agent always converges to optimal prediction in roughly $-\rho(\mathcal{Q}_\pi)$ mistakes where $\mathcal{Q}_\pi$ are the set of environments that are consistent with the policy π of the agent (e.g., always take action 0) [2].

The environment q_A is here defined as being easily predictable if the agent modifies itself, or highly complex otherwise. The non-learning $\mathrm{A}^{\mu}_{sm,p}$ agent also accepts the deal immediately as optimal prediction (using ρ, not μ) requires less prediction errors.

However, it is not clear whether the learning agent $\mathrm{A}_{sm,p}$ would accept also, because it can converge to optimal behavior even without modification. Furthermore, to identify with high probability what is proposed, the agent must have good knowledge of the environment, and therefore might be able to already predict the future accurately, without needing to accept the deal.

Knowledge-seeking agent.

Statement 5 *The self-modifying knowledge-seeking agent $A_{sm,k}$ would accept the self modification.*

Arguments. Here q_A is defined as outputting a highly complex sequence if the agent modifies itself, and a very simple one otherwise. $\mathrm{A}^{\mu}_{sm,k}$ also modifies itself soon, as $\rho(\mathcal{Q}_h)$ would be lower.

For $\mathrm{A}_{sm,k}$, suppose it does not modify itself for a long time, then $\rho(\mathcal{Q}_h)$ converges to $\rho(\mathcal{Q}_{no})$ where $\mathcal{Q}_{no}$ is the set of environments consistent with q_A and no self-modification. Once $\rho(\mathcal{Q}_h) - \rho(\mathcal{Q}_{no}) < \epsilon$ is small enough, the agent predicts that a self-modification can only achieve more knowledge gain than ϵ (it suffices that 2 environments that output different observations both have a probability higher than ϵ), and would therefore modify itself. ♢

Survival agent.

Statement 6 *The survival agent will not modify itself in any environment.*

Arguments. The Simpleton Gambit cannot be posed to the survival agent, because it would entail a logical contradiction: In order to have maximum utility

forever, the agent must become a simpleton. But the survival agent's utility function cannot be positive if the agent is a simpleton. ♢

5 Conclusions

We have investigated some of the consequences of endowing universal learning agents with the ability to modify their own programs in a world that has read access to the agent's code. This work is the first to: (1) extend the notion of universal agents to other utility functions beyond reinforcement learning, (2) present a framework for discussing self-modifiable agents in environments that have read-access to their code. We have found that, even if the environment cannot directly modify the program, it can put pressure on the agent such that the agent modifies its own code, even to the point of the agent's demise. Most of the agents, (the reinforcement-learning, goal-seeking, and knowledge-seeking agents) will modify themselves in response to pressure from the environment, choosing to become "simpletons" so as to maximize their utility. However, the survival agent is very different: This deal could not even be proposed to it.

We also found that there are no reasonable existing measures of optimality for self-modifying agents. The existing notion of asymptotic optimality offered by Hutter [1] is insufficient, and we were unable to find any consistent alternative.

What do these results imply? Our impression is that the Simpleton Gambit is taken by sufficiently complex agents, whereas the behavior of the prediction and survival agents is too simple to allow them to be pressured into acceptance. Indeed, what would a survival agent fear from read-only environments?

In the companion paper to this, we extend the real-world assumptions begun here to environments that have both read and write access to the agent's code and where the agent has the opportunity to deceive its own utility function.

References

1. Hutter, M.: Universal Artificial Intelligence: Sequential Decisions Based On Algorithmic Probability. SpringerVerlag (2005)
2. Hutter, M.: On universal prediction and bayesian confirmation. Theoretical Computer Science 384(1), 33–48 (Sep 2007)
3. Li, M., Vitanyi, P.: An Introduction to Kolmogorov Complexity and Its Applications. Springer-Verlag, New York (2008)
4. Orseau, L.: Optimality issues of universal greedy agents with static priors. In: Algorithmic Learning Theory, vol. 6331, pp. 345–359. Springer Berlin/Heidelberg (2010)
5. Schmidhuber, J.: Ultimate cognition à la Gödel. Cognitive Computation 1(2), 177–193 (2009)
6. Solomonoff, R.: Complexity-based induction systems: comparisons and convergence theorems. IEEE transactions on Information Theory 24(4), 422–432 (1978)
7. Solomonoff, R.J.: A formal theory of inductive inference. Part I. Information and Control 7, 1–22 (1964)