Vingean Reflection: Reliable Reasoning for Self-Improving Agents

Benja Fallenstein and **Nate Soares**
Machine Intelligence Research Institute
{benja,nate}@intelligence.org

Abstract

Today, human-level machine intelligence is in the domain of futurism, but there is every reason to expect that it will be developed eventually. Once artificial agents become able to improve themselves further, they may far surpass human intelligence, making it vitally important to ensure that the result of an "intelligence explosion" is aligned with human interests. In this paper, we discuss one aspect of this challenge: ensuring that the initial agent's reasoning about its future versions is reliable, even if these future versions are far more intelligent than the current reasoner. We refer to reasoning of this sort as *Vingean reflection*.

A self-improving agent must reason about the behavior of its smarter successors in abstract terms, since if it could predict their actions in detail, it would already be as smart as them. This is called the *Vingean principle*, and we argue that theoretical work on Vingean reflection should focus on formal models that reflect this principle. However, the framework of expected utility maximization, commonly used to model rational agents, fails to do so. We review a body of work which instead investigates agents that use formal proofs to reason about their successors. While it is unlikely that real-world agents would base their behavior entirely on formal proofs, this appears to be the best currently available formal model of abstract reasoning, and work in this setting may lead to insights applicable to more realistic approaches to Vingean reflection.

1 Introduction

In a 1965 article, I.J. Good introduced the concept of an "intelligence explosion" (Good 1965):

> Let an ultraintelligent machine be defined as a machine that can far surpass all the intellectual activities of any man however clever. Since the design of machines is one of these intellectual activities, an ultraintelligent machine could design even better machines; there would then unquestionably be an 'intelligence explosion,' and the intelligence of man would be left far behind. Thus the first ultraintelligent machine is the last invention that man need ever make.

Almost fifty years later, a machine intelligence that is smart in the way humans are remains the subject of futurism and science fiction. But barring global catastrophe, there seems to be little reason to doubt that humanity will *eventually* create a smarter-than-human machine. Whether machine intelligence can really leave the intelligence of biological humans far behind is less obvious, but there is some reason to think that this may be the case (Bostrom 2014): First, the hardware of human brains is nowhere close to physical limits; and second, not much time has passed on an evolutionary timescale since humans developed language, suggesting that we possess the *minimal* amount of general intelligence necessary to develop a technological civilization, not the theoretical optimum.

It's not hard to see that if building an artificial superintelligent agent will be possible at some point in the future, this could be both a great boon to humanity and a great danger if this agent does not work as intended (Bostrom 2014; Yudkowsky 2008). Imagine, for example, a system built to operate a robotic laboratory for finding a cure for cancer; if this is its only goal, and the system becomes far smarter than any human, then its best course of action (to maximize the probability of achieving its goal) may well be to convert all of Earth into more computers and robotic laboratories—and with sufficient intelligence, it may well find a way to do so. This argument generalizes, of course: While there is no reason to think that an artificial intelligence would be driven by human motivations like a lust for power, *any* goals that are not quite ours would place it at odds with our interests.

How, then, can we ensure that self-improving smarter-than-human machine intelligence, if and when

Research supported by the Machine Intelligence Research Institute (intelligence.org). Published as Technical report 2015–2.

it is developed, is beneficial to humanity?

Extensive testing may not be sufficient. A smarter-than-human agent would have an incentive to pretend during testing that its goals are aligned with ours, even if they are not, because we might otherwise attempt to modify it or shut it down (Bostrom 2014). Hence, testing would only give reliable information if the system is not yet sufficiently intelligent to deceive us. If, at this point, it is also not yet intelligent enough to realize that its goals are at odds with ours, a misaligned agent might pass even very extensive tests.

Moreover, the test environment may be very different from the environment in which the system will actually operate. It may be infeasible to set up a testing environment which allows a smarter-than-human system to be tested in the kinds of complex, unexpected situations that it might encounter in the real world as it gains knowledge and executes strategies that its programmers never conceived of.

For these reasons, it seems important to have a theoretical understanding of why the system is expected to work, so as to gain high confidence in a system that will face a wide range of unanticipated challenges (Soares and Fallenstein 2014a). By this we mean two things: (1) a formal specification of the problem faced by the system; and (2) a firm understanding of why the system (which must inevitably use practical heuristics) is expected to perform well on this problem.

It may seem odd to raise these questions today, with smarter-than-human machines still firmly in the domain of futurism; we can hardly verify that the heuristics employed by an artificial agent work as intended before we even know what these heuristics are. However, Soares and Fallenstein (2014a) argue that there is *foundational* research we can do today that can help us understand the operation of a smarter-than-human agent on an abstract level.

For example, although the expected utility maximization framework of neoclassical economics has serious shortcomings in describing the behavior of a realistic artificial agent, it is a useful starting point for asking whether it's possible to avoid giving a misaligned agent incentives for manipulating its human operators (Soares and Fallenstein 2015). Similarly, it allows us to ask what sorts of models of the environment would be able to deal with the complexities of the real world (Hutter 2000). Where this framework falls short, we can ask how to extend it to capture more aspects of reality, such as the fact that an agent is a part of its environment (Orseau and Ring 2012), and the fact that a real agent cannot be logically omniscient (Gaifman 2004; Soares and Fallenstein 2015). Moreover, even when more realistic models are available, simple models can clarify conceptual issues by idealizing away difficulties not relevant to a particular problem under consideration.

In this paper, we review work on one foundational issue that would be particularly relevant in the context of an intelligence explosion—that is, if humanity does not create a superintelligent agent directly, but instead creates an agent that attains superintelligence through a sequence of successive self-improvements. In this case, the resulting superintelligent system may be quite different from the initial verified system. The behavior of the final system would depend entirely upon the ability of the initial system to reason correctly about the construction of systems more intelligent than itself.

This is no trouble if the initial system is extremely reliable: if the reasoning of the initial agent were at least as good as a team of human AI researchers in all domains, then the system itself would be at least as safe as anything designed by a team of human researchers. However, if the system were only known to reason well in *most* cases, then it seems prudent to verify its reasoning specifically in the critical case where the agent reasons about self-modifications.

At least intuitively, reasoning about the behavior of an agent which is more intelligent than the reasoner seems qualitatively more difficult than reasoning about the behavior of a less intelligent system. Verifying that a military drone obeys certain rules of engagement is one thing; verifying that an artificial general would successfully run a war, identifying clever strategies never before conceived of and deploying brilliant plans as appropriate, seems like another thing entirely. It is certainly possible that this intuition will turn out to be wrong, but it seems as if we should at least *check*: if extremely high confidence must be placed on the ability of self-modifying systems to reason about agents which are smarter than the reasoner, then it seems prudent to develop a theoretical understanding of satisfactory reasoning about smarter agents. In honor of Vinge (1993), who emphasizes the difficulty of predicting the behavior of smarter-than-human agents with human intelligence, we refer to reasoning of this sort as *Vingean reflection*.

2 Vingean Reflection

The simplest and cleanest formal model of intelligent agents is the framework of expected utility maximization. Given that this framework has been a productive basis for theoretical work both in artificial intelligence in general, and on smarter-than-human agents in particular, it is natural to ask whether it can be used to model the reasoning of self-improving agents.

However, although it can be useful to consider models that idealize away part of the complexity of the real world, it is not difficult to see that in the case of self-improvement, expected utility maximization idealizes away too much. An agent that can literally maximize expected utility is *already* reasoning optimally; it may lack information about its environment, but it can only fix this problem by observing the external world, not by improving its own reasoning processes.

A particularly illustrative example of the mismatch between the classical theory and the problem of Vingean reflection is provided by the standard technique of *backward induction*, which finds the optimal policy of an agent facing a sequential decision problem by considering every node in the agent's entire decision tree. Backward induction starts with the leaves, figuring out the action an optimal agent would take in the last timestep (for every possible history of what happened in the previous timesteps). It then proceeds to compute how an optimal agent would behave in the second-to-last timestep, given the behavior in the last timestep, and so on backward to the root of the decision tree.

A self-improving agent is supposed to become more intelligent as time goes on. An agent using backward induction to choose its action, however, would have to compute its exact actions in every situation it might face in the future in the very first timestep—but if it is able to do that, its initial version could hardly be called less intelligent than the later ones!

Since we are interested in theoretical understanding, the reason we see this as a problem is not that backward induction is impractical as an implementation technique. For example, we may not actually be able to run an agent which uses backward induction (since this requires effort exponential in the number of timesteps), but it can still be useful to ask how such an agent would behave, say in a situation where it may have an incentive to manipulate its human operators (Soares and Fallenstein 2015). Rather, the problem is that we are trying to understand conceptually how an agent can reason about the behavior of a more intelligent successor, and an "idealized" model that requires the original agent to already be as smart as its successors seems to idealize away the very issue we are trying to investigate.

The programmers of the famous chess program Deep Blue, for example, couldn't have evaluated different heuristics by predicting, in their own heads, where each heuristic would make Deep Blue move in every possible situation; if they had been able to do so, they would have been able to play world-class chess themselves. But this does not imply that they knew nothing about Deep Blue's operation: their abstract knowledge of the code allowed them to know that Deep Blue was trying to win the game rather than to lose it, for example.

Like Deep Blue's programmers, any artificial agent reasoning about smarter successors will have to do so using abstract reasoning, rather than by computing out what these successors would do in every possible situation. Yudkowsky and Herreshoff (2013) call this observation the *Vingean principle*, and it seems to us that progress on Vingean reflection will require formal models that implement this principle, instead of idealizing the problem away.

This is not to say that expected utility maximization has no role to play in the study of Vingean reflection. Intuitively, the reason the classical framework is unsuitable is that it demands logical omniscience: It assumes that although an agent may be uncertain about its environment, it must have perfect knowledge of all mathematical facts, such as which of two algorithms is more efficient on a given problem or which of two bets leads to a higher expected payoff under a certain computable (but intractable) probability distribution. Real agents, on the other hand, must deal with *logical uncertainty* (Soares and Fallenstein 2015). But many proposals for dealing with uncertainty about mathematical facts involve assigning probabilities to them, which might make it possible to maximize expected utility with respect to the resulting probability distribution.

However, while there is some existing work on formal models of logical uncertainty (see Soares and Fallenstein [2015] for an overview), none of the approaches the authors are aware of are models of abstract reasoning. It is clear that any agent performing Vingean reflection will need to have some way of dealing with logical uncertainty, since it will have to reason about the behavior of computer programs it cannot run (in particular, future versions of itself). At present, however, formal models of logical uncertainty do not yet seem up to the task of studying abstract reasoning about more intelligent successors.

In this paper, we review a body of work which instead considers agents that use formal proofs to reason about their successors, an approach first proposed by Yudkowsky and Herreshoff (2013). In particular, following these authors, we consider agents which will only perform actions (such as self-modifications) if they can prove that these actions are, in some formal sense, "safe". We do not argue that this is a realistic way for smarter-than-human agents to reason about potential actions; rather, formal proofs seem to be the best formal model of abstract reasoning available at present, and hence currently the most promising vehicle for studying Vingean reflection.

There is, of course, no guarantee that results obtained in this setting will generalize to whatever forms of reasoning realistic artificial agents will employ. However, there is some reason for optimism: at least one such result (the *procrastination paradox* [Yudkowsky 2013], discussed in Section 4) both has an intuitive interpretation that makes it seem likely to be relevant beyond the domain of formal proofs, and has been shown to apply to one existing model of self-referential reasoning under logical uncertainty (Fallenstein 2014b).

The study of Vingean reflection in a formal logic framework also has merit in its own right. While formal logic is not a good tool for reasoning about a complex environment, it is a useful tool for reasoning about the properties of computer programs. Indeed, when humans require extremely high confidence in a computer program, they often resort to systems based on formal logic, such as model checkers and theorem provers (US DoD 1985; UK MoD 1991). Smarter-than-human machines attempting to gain high confidence in a computer program may need to use similar techniques. While smarter-than-human agents must ultimately reason under logical uncertainty, there is some reason to expect that high-confidence logically uncertain reasoning about computer programs will require something akin to formal logic.

The remainder of this paper is structured as follows. In the next section, we discuss in more detail the idea of requiring an agent to produce formal proofs that its actions are safe, and discuss a problem that arises in this context, the *Löbian obstacle* (Yudkowsky and Herreshoff 2013): Due to Gödel's second incompleteness theorem, an agent using formal proofs cannot trust the reasoning of future versions using the same proof system. In Section 4, we discuss the *procrastination paradox*, an intuitive example of what can go wrong in a system that trusts its own reasoning too much. In Section 5, we introduce a concrete toy model of self-rewriting agents, and discuss the Löbian obstacle in this context. Section 6 reviews partial solutions to this problem, and Section 7 concludes.

3 Proof-Based Systems

When humans want extremely high confidence in the properties of a computer program, they often resort to machine-checked proofs. Schmidhuber (2007) suggests a proof-based architecture for self-modifying artificial agents, called *Gödel machines*. These machines run both an initial problem solver, which determines the machine's interactions with the external world and has been verified by conventional methods, and a proof searcher, which attempts to find rewrites (of the initial problem solver, the proof searcher itself, or the runtime hosting both of these) that it can formally prove to be an improvement on the current version. (For details, see Schmidhuber [2007]. However, the issues of self-reference discussed in this paper appear to apply to the Gödel machine architecture as envisioned by Schmidhuber; see Fallenstein and Soares [2014].)

Yudkowsky and Herreshoff (2013) have proposed a related architecture in which a generally intelligent *suggester* handles both rewrites and interactions with the external world, but must prove that any action it chooses to perform fulfills a certain safety property. The action is only performed after this proof has been checked by a simple *verifier*. This leaves open what happens if the suggester is not able to find such a proof. In analogy with the Gödel machine's initial problem solver, in this case control can be diverted to a *fallback policy* which has been verified to satisfy the safety property, but which may otherwise be quite suboptimal (Fallenstein and Soares 2014).

This set-up is reminiscent of the *simplex architecture* of Seto et al. (1998), a software technology designed to allow the upgrade of a real-time system while it is running. A system employing the simplex architecture simultaneously runs two different controllers and a control switching logic. The first, *experimental* controller is allowed to run the system initially, but if the system goes outside a safety envelope, the control switching logic hands control over to the second, *baseline* controller. Like the experimental controller in the simplex architecture, the suggester is run in hopes that it will get better results than the fallback policy (which is analogous to the baseline controller). An acceptable baseline level of performance is guaranteed, because like the control switching logic, the verifier will divert control to the fallback policy if necessary.

However, unlike in the simplex architecture, we do not need to rely on *detecting* departures from a safety envelope, which may be difficult in the case of a generally intelligent system; instead, the suggester is responsible for convincing the verifier that its proposed plan of action satisfies the safety property. Moreover, the suggester is able to require future versions to prove an even stronger safety property, as long as it can also exhibit a new fallback policy that it can prove to satisfy this property; thus, the agent can ratchet up the effectiveness of its "baseline controller" over time.

Of course, constructing such an agent requires that we *have* formally defined safety properties that apply to the actions of an artificial agent. There are reasons to believe that developing such a safety predicate would be a very difficult task, but these issues are beyond the scope of the present paper; see Soares (2015) for discussion. Here, we assume that such a safety predicate is given, and focus on difficulties that arise when an agent with verifiable safety properties, using a suggester-verifier architecture, attempts to self-improve.

In particular, consider a suggester which is proposing a trivial self-modification, or even a suggester attempting to execute an action that will leave the agent unchanged. In order to prove that such an action is safe, it must show not only that the direct effects of this action satisfy the safety property, but also that all actions the modified agent will take in the future will satisfy this property as well.

Outside the system, *we* have concluded that the agent is safe by the following argument: either the agent takes an action that its suggester has proved to be safe, or it executes an action chosen by its fallback policy, whose safety has been separately verified; in either case, the action is safe. Intuitively, we expect that the agent itself should be able to follow a similar line of reasoning when reasoning about the safety of future versions of itself.

As Yudkowsky and Herreshoff (2013) point out, however, while the suggester can show that the new version will either execute a (safe) fallback policy or an action that it has proven safe, it cannot in general conclude that an action which has been *proven* safe is *in fact* safe. Intuitively, this follows from Gödel's second incompleteness theorem: No consistent formal system as strong as Peano Arithmetic is able to prove its own consistency. Thus, any first-order theory T our agent might use must have non-standard models that believe T to be inconsistent (i.e., models in which the sentence $\mathrm{Con}(T)$, which asserts T's consistency, is false). But in such a model, the fact that an action is provably safe clearly does not in general imply that this action is in fact safe, since if T is inconsistent, *all* actions are provably safe. More formally, according to Löb's theorem (Löb 1955) (which generalizes the second incompleteness theorem), T can only prove a sentence of the form "if T proves φ, then φ" for formulas φ which it already proves unconditionally; thus, T cannot prove that any provably safe action is in fact safe unless it can prove all actions safe.

This may seem like a rather abstract problem, traceable to a quirk of formal logic. However, there is some reason to think that it may apply far more generally. We illustrate this in the next section by the *procrastination paradox* (Yudkowsky 2013; Fallenstein and Soares 2014), a concrete way in which an agent which trusts future versions too much may go wrong. When our agents use formal logic, this is very similar to Gödel's proof, but the informal argument seems likely to apply to other approaches as well.

4 The Procrastination Paradox

Consider a suggester-verifier agent with a very simple goal: ensuring that a certain button gets pressed, either now or at some point in the future. Suppose further that time is divided into infinitely many discrete timesteps, and the agent must choose between only two options; it can either press the button in the current timestep, or leave everything unchanged, in which case it will be in exactly the same state and faced with exactly the same choice at the beginning of the next timestep.

Intuitively, it is clear that in order to achieve its goal, the agent must press the button immediately—otherwise, since the next timestep is exactly the same, it will again decide to "procrastinate," and so on in all future timesteps. We now show how an agent that trusts future versions too much can argue itself into believing the opposite.

Write $\square_T\ulcorner\varphi\urcorner$ for the proposition that the sentence φ is provable in the theory T. For any recursively enumerable theory T, define the *uniform reflection principle* $\mathrm{REF}(T)$ to be the collection of all sentences of the form

$$\forall n.\ \square_T\ulcorner\varphi(\overline{n})\urcorner \to \varphi(n), \tag{1}$$

where $\varphi(\cdot)$ ranges over formulas with one free variable in the language of arithmetic, and $\overline{n}$ indicates the numeral corresponding to n; e.g., if $n = 2$, then $\ulcorner\varphi(\overline{n})\urcorner$ is the Gödel number of the sentence $\varphi(S(S(0)))$.

Assume that our agent is using a particular theory T extending Peano Arithmetic (PA) and write G for the proposition that the button is pressed sometime after the current timestep. Our suggester attempts to prove that G is true, in which case it "procrastinates"; otherwise, our agent executes its fallback policy, which is to press the button immediately. If T can prove this by inspection of the agent's source code, then $T \vdash \neg G \to \square_T\ulcorner G\urcorner$; although we do not need to do so, if we assume that the suggester will find a proof if there is one,[1] then the implication turns into an equivalence, and G turns into the Gödel sentence "I am not provable" ($T \vdash G \leftrightarrow \neg\square_T\ulcorner G\urcorner$). As in Gödel's first incompleteness theorem, if T is sound, G must be true but unprovable: The agent must press the button immediately.

However, if T proves its own reflection principle (i.e., if $T \vdash \mathrm{REF}(T)$), our agent can also reason as follows: "Either I will press the button in the next timestep or I won't" ($T \vdash G \vee \neg G$). "The only case in which I would *not* press the button is if I have found a proof that the button gets pressed at a later time" ($T \vdash \neg G \to \square_T\ulcorner G\urcorner$). "If it's provable that I will press the button, then by (1), I will indeed do so" ($T \vdash \square_T\ulcorner G\urcorner \to G$). "Thus, whether or not I press the button in the next timestep, it eventually gets pressed" ($T \vdash G$); "hence, I do not need to press it now!"

Since $T \vdash G$ implies $T \vdash \square_T\ulcorner G\urcorner$ (if a recursively enumerable theory T proves a sentence φ, then any theory as strong as PA proves $\square_T\ulcorner\varphi\urcorner$), we also have $T \vdash \neg G$, implying that T is inconsistent; although it makes stronger assumptions, this is very similar in flavor to the proof of the second incompleteness theorem. However, the informal version of the agent's reasoning uses no special properties of first-order logic, and suggests that similar diagonalization issues may affect many other approaches to Vingean reflection. In fact, Fallenstein (2014b) shows that this is indeed a problem for a formalism for self-reference in logical uncertainty proposed by Christiano et al. (2013), which uses probabilities rather than formal proofs.

5 Agents in Botworld

The fact that requiring $T \vdash \mathrm{REF}(T)$ leads to inconsistency implies that the most straight-forward way of formalizing the intuition why a suggester should be able to justify simple self-modifications does not work; to model Vingean reflection with the suggester-verifier architecture, the proof of safety must use an alternate path.

In order to do so, it will be helpful to have a more formal model of the problem. In this section, we introduce two tools that have been developed for this purpose: A formalism that agents using the suggester-verifier architecture can use to reason about the decision problem they are facing (Fallenstein and Soares 2014), based on the "space-time embedded intelligence" model of Orseau and Ring (2012); and *Botworld* (Soares and Fallenstein 2014b), a concrete toy environment exhibiting some features of the real world that satisfactory Vingean reflection should be able to handle.

Botworld is structured as a cellular automaton. Each cell may contain multiple *items* and *robots*. The items come in many varieties, including *cargo*, which the robots are trying to collect for its intrinsic value, and *robot parts*, which may be used to create robots.

[1] This requires a halting oracle.

Each robot houses a simple register machine, which is run during each timestep in order to determine the robot's *action* in that timestep. Robots may manipulate (lift, drop, and carry) items. They may also create new robots (from component parts), destroy robots (into component parts), and inspect robots (reading the states of their register machines).

Consider "games" where the initial state of the automaton is fixed except for the programming of one particular robot. The task is to choose the program of this robot in such a way that a particular objective will be met (perhaps after the initial program has undergone many self-improvements). This provides a fairly flexible environment in which to study agents performing Vingean reflection. This paper lacks the space for a full description of Botworld, but further details and a complete, executable specification are given in Soares and Fallenstein (2014b).

To illustrate the Löbian obstacle, consider a simple *bodyguard game*, in which there are (at least) two robots: The *bodyguard robot* is the agent attempting to self-improve. The *assignment robot* is a robot that must not be destroyed. As long as the assignment survives, the player gets points for any cargo the bodyguard robot collects. In simple versions of this game, the player need only ensure that the bodyguard never unwittingly destroys the assignment. In more complex variants, other robots which attempt to destroy the assignment robot may be included in the environment.

We now introduce a simplified version of the formalism proposed by Fallenstein and Soares (2014). Instead of considering a sequence of interactions with the environment (such as considered by Orseau and Ring [2012]), this formalism focuses on a single choice: An agent makes a single observation $o \in \mathcal{O}$ and chooses a single action $a \in \mathcal{A}$. In Botworld, the observation is the input that the agent receives from the environment, and the action includes both the robots's behavior and the new state of its register machine. (Thus, decisions to self-modify and decisions *not* to self-modify are treated in exactly the same way.)

Define a *policy* to be a function $\pi \in \Pi$ which specifies, for every observation $o \in \mathcal{O}$, the action $\pi(o) \in \mathcal{A}$ that the agent will take if it makes this observation. Assume that there is a set $\mathcal{Q}$ of "outcomes" describing all features of the world, at all future times, that matter to the agent's goals. In the bodyguard game, set $\mathcal{Q} := (\{0,1\} \times \mathbb{N})^{\mathbb{N}}$, where for $q \in \mathcal{Q}$, the first element of $q(n)$ is 1 if the assignment robot is still "alive" in the n'th timestep, and the second element is the total value of all cargo held by the bodyguard robot at this time. Each agent is equipped with an "environment function" $Q : \mathcal{O} \times \mathcal{A} \to \mathcal{Q}$ which maps observations and actions onto outcomes such that $Q(o, a)$ denotes the future state achieved after making observation o and executing action a. This function must be fully specified, but *need not be feasibly computable*: the agent will *prove* things about the outcome its actions lead to, but will not look into the future to determine the precise outcome achieved. (The full formalism of Fallenstein and Soares (2014) allows the agent to have uncertainty, in the form of a probability distribution, over the results of its actions.)

Assume further that there is a *safety predicate* $\mathrm{Safe}(q)$, defined on outcomes $q \in \mathcal{Q}$, which specifies the safety property the agent's actions must satisfy. In the bodyguard game, this predicate is

$$\mathrm{Safe}(q) :\leftrightarrow \forall n.\ \exists x.\ q(n) = (1, x),$$

which simply states that the assignment robot is alive in all future timesteps.

Consider agents running a suggester-verifier architecture with a *fallback policy* $\pi_{\mathrm{fb}} \in \Pi$ which specifies a fallback action $\pi_{\mathrm{fb}}(o)$ to be taken after making observation o. This fallback action will be executed if the agent cannot find another provably safe action to take, and thus the player must verify $\mathrm{Safe}(Q(o, \pi_{\mathrm{fb}}(o)))$ for all o themselves (in order to verify that the system is safe). In the simple versions of the bodyguard game (where there are no robots besides the bodyguard and the assignment, and the assignment never attempts to deconstruct itself), this can simply be the policy which executes the `Pass` command in all future timesteps, thus never taking any external action at all.

An agent running a suggester-verifier architecture also contains an untrusted suggester σ, another computable function. This suggester may have some motivation, such as collecting high-value cargo, but cannot be trusted to suggest only safe actions. For example, σ may notice that the assignment robot is built from very expensive parts, and may suggest destroying the assignment robot and collecting its components—an action the verifier should not allow. Formally, the suggester takes an observation $o \in \mathcal{O}$ and attempts to return an action $a \in \mathcal{A}$ together with a formal proof of the target sentence $\ulcorner \mathrm{Safe}(Q(\overline{o}, \overline{a})) \urcorner$ in some formal theory T such as $T = \mathsf{PA}$, where $\overline{o}$ and $\overline{a}$ are the numerals denoting o and a. If no such action/proof pair can be found, the suggester returns an error value.

The verifier runs $\sigma(o)$ and, if this returns a pair (a, p) within the allotted time limit, checks whether p is a valid proof of the target sentence from the axioms of T. If so, it executes action a; otherwise, action $\pi_{\mathrm{fb}}(o)$. Denote this policy for selecting actions by $\mathrm{sv}_T(\ulcorner \mathrm{Safe} \urcorner, \ulcorner Q \urcorner, \sigma, \pi_{\mathrm{fb}}) \in \Pi$. (Here, we write $\ulcorner \mathrm{Safe} \urcorner$ and $\ulcorner Q \urcorner$ in Gödel quotes because the suggester-verifier only makes use of their definitions in first-order logic, while σ and π_{fb} need to be provided as runnable code.) The question is, can such an agent justify performing a self-modification?

Consider an agent as described above which has made some specific observation $\hat{o} \in \mathcal{O}$ and which is considering some specific action $\hat{a} \in \mathcal{A}$, which leaves the agent in a state such that, in the next step, it will execute the policy $\hat{\pi}' = \mathrm{sv}_{T'}(\ulcorner \mathrm{Safe}' \urcorner, \ulcorner Q' \urcorner, \sigma', \pi'_{\mathrm{fb}}) \in \Pi$ for some theory T', safety predicate Safe', outcome calculator Q', suggester σ', and fallback policy π'_{fb}. Obviously, not all such policies are safe (consider the policy where $\text{Safe}'(q) :\leftrightarrow (0 = 0)$). But if the policy *is* safe—say, if $T = T'$, $\text{Safe} = \text{Safe}'$ and π'_{fb} is provably safe on all inputs—then will the current suggester σ be able to exhibit a proof of this fact?

In order to prove the safety of $\hat{\pi}'$, the agent must first prove that the new agent's safety predicate is at least as strong as the existing safety predicate. Thus, it must be the case that:

$$T \ \vdash \ \forall q \in \mathcal{Q}.\, \text{Safe}'(q) \rightarrow \text{Safe}(q). \tag{2}$$

Note that it is fine for the predicate Safe' to be *more* restrictive than Safe: for example, it could be that $\text{Safe}'(q) :\leftrightarrow \forall n.\ (\exists x.\ q(n) = (1, x) \land x \geq 7)$, which ensures that the assignment robot survives and that the agent scores at least 7 points.

Of course, this property only matters if the successor agent accurately assesses world outcomes. The safety predicate Safe' is of no consequence if $Q'(o, a) = (1, 1000)$ unconditionally. Thus, the agent must also prove

$$T \ \vdash \ \exists o' \in \mathcal{O}.\, Q'(o', \hat{\pi}'(o')) \ = \ Q(\overline{\hat{o}}, \overline{\hat{a}}) \tag{3}$$

which intuitively states that Q' agrees with Q about world outcomes.

Next, the agent must be able to show that the fallback policy π'_{fb} of $\hat{\pi}'$ is safe:

$$T \ \vdash \ \forall o' \in \mathcal{O}.\, \text{Safe}'(Q'(o', \overline{\pi'_{\text{fb}}}(o'))) \tag{4}$$

By the implementation of the suggester-verifier architecture, this allows our agent to reason that executing action $\hat{a}$ (and thus adopting policy $\hat{\pi}'$) will *either* result in the execution of $\pi'_{\text{fb}}(o')$ (which, by (4), is safe), *or* of an action a' such that it is provable in the theory T' that $\ulcorner \text{Safe}'(Q'(\overline{o'}, \overline{a'})) \urcorner$. Thus, if $T \vdash \text{REF}(T')$, then our agent can prove that executing the aciton $\hat{a}$ is safe, as desired. (Fallenstein and Soares [2014] discuss this proof in detail.) But the last condition requires that $T' \neq T$: otherwise, T will be inconsistent and our agent would be able to prove any action "safe," for the reasons described in Section 4.

In the next section, three partial solutions to the Löbian obstacle are discussed.

6 Partial Solutions

6.1 Descending Trust

One obvious way to address this problem of self-reference is to require that the child agent use a weaker system for verifying proofs than the parent agent (Yudkowsky and Herreshoff 2013). If this allows the parent agent to verify the consistency and soundness of the child agent's proof system, then the parent agent can license the creation of a safe child. We call this the *descending trust* method of Vingean reflection.

Consider a parent agent using the theory $T := \mathsf{PA} + \text{REF}(\mathsf{PA})$. If the child agent $\hat{\pi}'$ under consideration uses the theory $T' = \mathsf{PA}$ to verify its proofs, then $T \vdash \text{REF}(T')$. This, along with (2), (3) and (4), allows the parent agent to prove its target theorem and license the action $\hat{a}$ which constructs the agent $\hat{\pi}'$. But if the child agent wants to license the creation of another agent, it will be in the same situation as if T had been PA to start with.

This issue can be alleviated by using a tower of theories of increasing strength. For example, an agent using the theory $\mathsf{PA} + \text{REF}(\mathsf{PA} + \text{REF}(PA))$ may license the construction of an agent using $\mathsf{PA} + \text{REF}(\mathsf{PA})$, which can license the construction of an agent using PA. Using the notation $T + 1$ to denote $T + \text{REF}(T)$, it is easy to see that an agent verifying proofs using $\mathsf{PA} + n$ may license descendant chains up to depth n before being forced to construct successor agents that verify proofs with a theory weaker than PA.

Thus, for tasks that require only a finite number of timesteps, it may suffice to build agents with verifier components using the theory $\mathsf{PA} + n$ for some absurdly large n. Instead of using finite n, any computable ordinal α may also be used. (Such "transfinite progressions" have been studied by Turing [1939] and Feferman [1962].) Unfortunately, by the well-foundedness of the ordinals, this method can still only license the creation of finite chains of successors.

Even if this method can be employed for finite tasks, the result is somewhat unsatisfacotry. These agents, which are ostensibly self-modifying in attempts to self-*improve*, can only verify the construction of agents which use *weaker* proof systems in their verifier component.

6.2 Herreshoff's Waterfall

One might hope that there exists some infinite sequence of theories T_n such that each T_n proves $\text{REF}(T_{n+1})$. Unfortunately, any such sequence of recursively axiomatizable theories must be unsound, by a variant of the procrastination paradox discussed in Section 4 (Yudkowsky 2013). However, it turns out to be possible to find a sequence T_n such that $T_n \vdash \text{Con}(T_{n+1})$ for every n.

Even this requires care. Naïvely, one might attempt to define the sequence by $T_n^{\text{bad}} := \mathsf{PA} + \text{Con}(T_{n+1}^{\text{bad}})$ (that is, where each theory is constructed from PA and axioms which state that the next theory in the sequence is consistent—the recursive definition can be made formal by using the diagonal lemma to define a primitive recursive predicate $\varphi(n, m)$, which is true if m is the Gödel number of an axiom of T_n). However, it turns out that every theory T_n^{bad} defined in this way is inconsistent (Stiennon 2013).

Instead, consider the set of theories

$$T_n := \mathsf{PA} + (\psi(n) \rightarrow \text{Con}(T_{n+1})), \tag{5}$$

where $\psi(n)$ is true if n is *not* the Gödel number of a proof of an inconsistency from the axioms of ZFC.

Then, although each T_n proves $\psi(n)$ (assuming that ZFC is consistent) and therefore $\mathrm{Con}(T_{n+1})$, this fact is not provable in PA, and it turns out that because of this, all T_n are consistent (Fallenstein 2014a) (and therefore sound, since the only axiom of T_n that is not an axiom of PA merely asserts the consistency of T_{n+1}). This idea, due to Marcello Herreshoff, was originally described by Yudkowsky and Herreshoff (2013), and the version presented here was proposed by Fallenstein (2014a).

It is well-known that if a theory T extends PA and if $T \vdash \mathrm{Con}(T')$, then T proves the reflection principle restricted to Π_1 formulas (recall that a formula $\varphi(n)$ is Π_1 if it is provably equivalent to $\forall m.\ \tilde{\varphi}(m,n)$, where $\tilde{\varphi}(m,n)$ is some predicate that can be decided by a primitive recursive function); i.e.,

$$T \vdash \forall n.\ \Box_{T'} \ulcorner \varphi(\overline{n}) \urcorner \to \varphi(n)$$

for all $\varphi \in \Pi_1$. Thus, so long as the safety predicate Safe that an agent must prove in order to construct its successor agent is a Π_1 sentence, a suggester-verifier agent using T_n may license the creation of a successor agent using T_{n+1}.

This result is heartening, if not entirely satisfactory. Agents reasoning using the theory T_n can trust the next theory T_{n+1}, and can indeed trust the theory T_{n+m} for any finite m (so long as they can prove $\psi(n+m)$), but such agents *cannot* conclude that $\forall m.\ \psi(n+m)$. Thus, such agents must entertain the possibility that they will one day run out of ability to verify new agents. We, outside the system, know that they never will, but they, inside the system, cannot show this fact. This seems like a strange epistemic state for a rational agent to occupy.

6.3 Model Polymorphism

It seems odd to demand that the theory used by the agent's verifier component must change with every timestep. Is there some assumption that could be relaxed in order to allow the agent to safely use the same theory in each timestep?

It turns out there is. One approach for doing so is *model polymorphism* (see Yudkowsky and Herreshoff 2013, where it is referred to as *parametric polymorphism*). This approach can be implemented by a theory MP in the language of PA augmented by a new constant symbol κ, which consists of the axioms of PA (with the axiom schema of induction ranging over formulas in the extended language) extended by all sentences of the form

$$\kappa > 0 \to \forall n.\ \big(\Box_{\mathsf{MP}} \ulcorner \varphi(\overline{n}) \urcorner \to \varphi[\kappa := \kappa - 1](n)\big), \quad (6)$$

where $\varphi(\cdot)$ ranges over one-variable formulas in the extended language, and $\varphi[\kappa := \kappa - 1]$ denotes the formula obtained by replacing all occurrences of the constant κ in φ by the term $\kappa - 1$.

Model polymorphism requires a modification to the suggester-verifier architecture in order to work. Instead of requiring a proof that an action is safe for all future times, require instead that an action is safe "for the next κ timesteps". Outside the system, we can show that $(\mathbb{N}, \kappa = n) \vDash \mathsf{MP}$ for any n, i.e., that the theory MP is sound if κ is interpreted as any natural number. Thus, we can be confident that the system is safe for an arbitrary number of steps. The system itself cannot reason in this way, and thereby avoids the standard Gödelian difficulties. The proof of soundness begins by noting that $(\mathbb{N}, \kappa = 0)$ trivially models MP (as its special axiom schema is vacuously true in this case), and then proceeds to show by induction that $(\mathbb{N}, \kappa = n) \vDash \mathsf{MP}$ implies $(\mathbb{N}, \kappa = n+1) \vDash \mathsf{MP}$; for details, see Yudkowsky and Herreshoff (2013).

MP requires replacing the unary predicate $\mathrm{Safe}(q)$ (which verifies a safety property in all future timesteps) by a binary predicate $\mathrm{Safe}(n,q)$ (which verifies a safety property in the n^{th} timestep). In the bodyguard game, this is the predicate $\mathrm{Safe}(n,q) :\leftrightarrow \exists x.\ q(n) = (1,x)$, which states that the assignment robot is still alive in the n'th timestep.

An agent using MP which can perform Vingean reflection can be constructed as follows: first, the agent must know the current timestep t. Its target theorem for the action a is

$$\ulcorner \forall n \le \bar{t} + \kappa.\ \mathrm{Safe}(n, Q(\overline{o}, \overline{a})) \urcorner.$$

Then, in any universe with discrete timesteps, a parent agent verifying proofs with MP may license the creation of a successor agent also running MP. The trick is that the parent agent need only prove that the child agent executes safe actions for $\kappa - 1$ steps, as the child agent won't begin running until the following timestep.

The suggester must prove slightly modified versions of (2) and (4), with the new Safe predicates:

$$\mathsf{MP} \vdash \forall n.\ \forall q \in \mathcal{Q}.\ \mathrm{Safe}'(n,q) \to \mathrm{Safe}(n,q) \quad (7)$$

$$\mathsf{MP} \vdash \forall n \le \bar{t} + \kappa.\ \forall o' \in \mathcal{O}.\ \mathrm{Safe}'(n, Q'(o', \overline{\pi'_{\mathrm{fb}}}(o'))) \quad (8)$$

Then the agent will be able to reason (by inspection of the suggester-verifier architecture) that either the successor will execute its fallback policy (which is safe for at least κ steps by (8)) or will execute some other action a' after proving in theory T' that $\ulcorner \forall n < \overline{t+1} + \kappa.\ \mathrm{Safe}'(n, Q'(\overline{o'}, \overline{a'})) \urcorner$.

Now, if $T' = \mathsf{MP}$ then the suggester can use the MP axiom schema to deduce

$$\forall n < \overline{t+1} + \kappa - 1.\ \mathrm{Safe}'(n, Q'(\overline{o'}, \overline{a'})), \quad (9)$$

which it can rewrite to

$$\forall n < \bar{t} + \kappa.\ \mathrm{Safe}'(n, Q'(\overline{o'}, \overline{a'})). \quad (10)$$

This, along with (7) and (3) gives us the target theorem, as desired.

An agent reasoning using model polymorphism can license the construction of another agent using the same theory. Unlike Herreshoff's waterfall, such agents can reason that they will be able to continue licensing the creation of new safe agents indefinitely.

However, it is difficult to interpret proofs about κ intuitively as arguments made by a rational agent; it is hard to make intuitive sense of the epistemic state of an agent that has proven a certain action to be safe for "κ" timesteps. In addition, model polymorphism only works in systems where time can be divided into discrete steps. While model polymorphism seems like a significant step forward, it is by no means a fully satisfactory solution.

7 Discussion

In this paper, we have discussed a pair of obstacles to Vingean reflection that arise in the context of proof-based suggester-verifier systems: the *Löbian obstacle* and the *procrastination paradox*. In order to create an agent that can self-modify with sufficient foresight to navigate an intelligence explosion, it seems that we will have to chart a course between the Löbian obstacle's rock of having so little self-trust that the agent cannot execute simple self-modifications, and the procrastination paradox's hard place of having so much self-trust that the agent's reasoning becomes unsound.

In the form we have discussed them, the Löbian obstacle and the procrastination paradox stem from the demand for *proofs* of safety; this is of course unrealistic in practice: an agent operating with uncertainty cannot realistically be expected to formally prove that specific action leads to an outcome with desirable properties. (However, Fallenstein [2014b] shows that a version of the procrastination paradox applies to the self-referential formalism for logical uncertainty proposed by Christiano et al. [2013].)

And yet, it intuitively seems that a system should be able to observe that another system using the same reasoning process has concluded φ and, from this, conclude φ: if reasoning of the form "that system reasons as I do and deduced φ, therefore φ" cannot be formalized by arguments of the form "a system using the same theory as me proved it, and therefore it is true," then how *can* this intuitively desirable reasoning be formalized?

One path is to relax the condition for action: instead of considering agents that execute action a only after proving that a is safe, allow agents to execute a if they can prove that a is safe *or* if they can prove that systems of similar strength prove a is safe *or* if they can prove that systems of similar strength prove that systems of similar strength prove a is safe, and so on. This is similar to the approach taken by Weaver (2013). However, this still does not allow an agent to execute a if it knows that another agent in the same situation, which is using the same condition for action, has taken action a: the fact that the agent knows $\exists n.\ \Box^n\ulcorner\varphi\urcorner$ does not allow it to conclude $\Box^{\overline{n}}\ulcorner\varphi\urcorner$ for any concrete n.

Another more generic solution would be to develop some sort of non-monotonic self-trust, allowing an agent to trust its own reasoning only in non-paradoxical cases, in the same way that the reader can trust their own reasoning without trusting their belief about the sentence "the reader does not believe this very sentence." However, it is not yet clear how to construct a reasoning system which has non-monotonic self-trust in this way.

The authors hope that continued study of Vingean reflection in proof-based models will shed light on how to move forward in one of these directions.

One topic for future work is to push the proof-based models towards more realistic settings. Fallenstein and Soares (2014) present a version of the suggester-verifier formalism in which agents maintain a probability distribution over external states; Yudkowsky (2014) and Soares (2014) discuss versions of this in the setting of dynamic Bayesian networks.

A second topic is to address the fact that the suggester-verifier agent formalism described in this paper only considers *satisficing* agents, in the sense of Simon (1956). The suggester in a suggester-verifier architecture is not required to prove that its chosen action is *optimal*; rather, it merely needs to find an action that is *good enough* (in the sense that it satisfies some goal predicate). There is a considerable gap between this kind of agent and a real-world agent that acts as intended: Suppose, for example, that an agent is constructed which satisfices its goals if it saves 3 billion lives in expectation. The agent discovers an asteroid en route to Earth which will end 7 billion lives if not stopped. This agent might find a plan that definitely stops the asteroid and then decide to flip a coin and deploy the plan if the coin lands heads: this gamble saves 3.5 billion lives in expectation, and thereby satisfies the agent's goal.

Intuitively, there should be some way to "boundedly maximize" expected utility, picking the best action available given the cost of spending more resources searching for better actions. "Staged maximization" (Yudkowsky 2014) is an attempt at beginning to formalize this intuition.

A *staged maximizer* divides the entire action space into categories and considers those. This idea is explored by Yudkowsky (2014), who discusses a number of new hurdles related to staged maximization.

To illustrate, consider a parent agent which must select an action from the set $\{\,\texttt{continue}, \texttt{stop}\,\}$. If the parent selects `continue` then the child will select an action from the set $\{\,a_1, a_2 \ldots, a_9\,\}$. If the parent selects `stop` then the child must select the action $\{\,\texttt{stopped}\,\}$. Each of the ten child actions has an associated true utility, and the parent makes an unbiased noisy estimate of each true utility. Say the true utility of each action $a_1 \ldots a_9$ is 0, the true utility of `stopped` is 0.1, and the noise is drawn from a normal distribution with mean 0 and standard deviation 1. In this case, the parent is biased towards `continue`, because that action leads to more possibilities, which leads to more opportunities for the unbiased noise estimates to be higher: the parent, if implemented naïvely, steers towards noise. This is known as the "optimizer's curse" (Smith and Winkler 2006), and it indicates that further research is necessary to better understand satisfactory staged maximization.

Regardless of what direction is taken, further research into Vingean reflection seems important. It may not be possible to delegate the study of Vingean reflection to self-modifying systems unless those systems are already at the level of a team of human researchers in all relevant regards, and it is plausible that significant self-modification (and, therefore, Vingean reflection) may be required to get to that point: Just as computer systems can be proficient at chess without being generally intelligent, it is plausible that a computer system could become proficient at some types of reasoning before becoming proficient at the kind of mathematical philosophy necessary to develop high-confidence methods for Vingean reflection.

Satisfactory models of Vingean reflection will have to look very different from the models presented in this paper: they will have to allow for logical uncertainty and could plausibly require non-monotonic self-trust. Nevertheless, study has to start somewhere, and we expect that starting with these highly simplified models and pushing them toward practicality could reveal more plausible paths toward practical methods for reliable Vingean reflection.

References

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

Christiano, Paul F., Eliezer Yudkowsky, Marcello Herreshoff, and Mihaly Barasz. 2013. *Definability of Truth in Probabilistic Logic.* Working Paper. Machine Intelligence Research Institute, Berkeley, CA, April 2. `https://intelligence.org/files/DefinabilityTruthDraft.pdf`.

Fallenstein, Benja. 2014a. *An Infinitely Descending Sequence of Sound Theories each Proving the Next Consistent.* Brief Technical Note. Machine Intelligence Research Institute, Berkeley, CA. `http://intelligence.org/files/ConsistencyWaterfall.pdf`.

———. 2014b. *Procrastination in Probabilistic Logic.* Working Paper. Machine Intelligence Research Institute, Berkeley, CA. `http://intelligence.org/files/ProbabilisticLogicProcrastinates.pdf`.

Fallenstein, Benja, and Nate Soares. 2014. "Problems of Self-Reference in Self-Improving Space-Time Embedded Intelligence." In *Artificial General Intelligence: 7th International Conference, AGI 2014, Quebec City, QC, Canada, August 1–4, 2014. Proceedings,* edited by Ben Goertzel, Laurent Orseau, and Javier Snaider, 21–32. Lecture Notes in Artificial Intelligence 8598. New York: Springer. doi:`10.1007/978-3-319-09274-4_3`.

Feferman, Solomon. 1962. "Transfinite Recursive Progressions of Axiomatic Theories." *Journal of Symbolic Logic* 27 (3): 259–316.

Gaifman, Haim. 2004. "Reasoning with Limited Resources and Assigning Probabilities to Arithmetical Statements." *Synthese* 140 (1–2): 97–119. doi:`10.1023/B:SYNT.0000029944.99888.a7`.

Good, Irving John. 1965. "Speculations Concerning the First Ultraintelligent Machine." In *Advances in Computers,* edited by Franz L. Alt and Morris Rubinoff, 6:31–88. New York: Academic Press. doi:`10.1016/S0065-2458(08)60418-0`.

Hutter, Marcus. 2000. "A Theory of Universal Artificial Intelligence based on Algorithmic Complexity." Unpublished manuscript, April 3. `http://arxiv.org/abs/cs/0004001`.

Löb, M. H. 1955. "Solution of a Problem of Leon Henkin." *Journal of Symbolic Logic* 20 (2): 115–118. `http://www.jstor.org/stable/2266895`.

Orseau, Laurent, and Mark Ring. 2012. "Space-Time Embedded Intelligence." In *Artificial General Intelligence: 5th International Conference, AGI 2012, Oxford, UK, December 8–11, 2012. Proceedings,* edited by Joscha Bach, Ben Goertzel, and Matthew Iklé, 209–218. Lecture Notes in Artificial Intelligence 7716. New York: Springer. doi:`10.1007/978-3-642-35506-6_22`.

Schmidhuber, Jürgen. 2007. "Gödel Machines: Fully Self-Referential Optimal Universal Self-Improvers." In *Artificial General Intelligence,* edited by Ben Goertzel and Cassio Pennachin, 199–226. Cognitive Technologies. Berlin: Springer. doi:`10.1007/978-3-540-68677-4_7`.

Seto, Danbing, Bruce Krogh, Lui Sha, and Alongkrit Chutinan. 1998. "The Simplex Architecture for Safe Online Control System Upgrades." In *Proceedings of the American Control Conference, 1998,* 6:3504–3508. IEEE.

Simon, Herbert A. 1956. "Rational Choice and the Structure of the Environment." *Psychological Review* 63 (2): 129.

Smith, James E., and Robert L. Winkler. 2006. "The Optimizer's Curse: Skepticism and Postdecision Surprise in Decision Analysis." *Management Science* 52 (3): 311–322. doi:`10.1287/mnsc.1050.0451`.

Soares, Nate. 2014. *Tiling Agents in Causal Graphs.* Technical report 2014–5. Berkeley, CA: Machine Intelligence Research Institute. `http://intelligence.org/files/TilingAgentsCausalGraphs.pdf`.

———. 2015. *The Value Learning Problem.* Technical report 2015–4. Berkeley, CA: Machine Intelligence Research Institute. `https://intelligence.org/files/ValueLearningProblem.pdf`.

Soares, Nate, and Benja Fallenstein. 2014a. *Aligning Superintelligence with Human Interests: A Technical Research Agenda.* Technical report 2014–8. Berkeley, CA: Machine Intelligence Research Institute. `https://intelligence.org/files/TechnicalAgenda.pdf`.

Soares, Nate, and Benja Fallenstein. 2014b. *Botworld.* Technical report 2014–2. Berkeley, CA: Machine Intelligence Research Institute. `http://intelligence.org/files/Botworld.pdf`.

———. 2015. *Questions of Reasoning Under Logical Uncertainty.* Technical report 2015–1. Berkeley, CA: Machine Intelligence Research Institute. `https://intelligence.org/files/QuestionsLogicalUncertainty.pdf`.

Stiennon, Nisan. 2013. *Recursively-Defined Logical Theories are Well-Defined.* Brief Technical Note. Berkeley, CA: Machine Intelligence Research Institute. `http://intelligence.org/files/RecursivelyDefinedTheories.pdf`.

Turing, Alan Mathison. 1939. "Systems of logic based on ordinals." *Proceedings of the London Mathematical Society* 2 (1): 161–228.

United Kingdom Ministry of Defense. 1991. *Requirements for the Procurement of Safety Critical Software in Defence Equipment.* Interim Defence Standard 00-55. United Kingdom Ministry of Defense, April 5.

United States Department of Defense. 1985. *Department of Defense Trusted Computer System Evaluation Criteria.* Department of Defense Standard DOD 5200.28-STD. United States Department of Defense, December 26. `http://csrc.nist.gov/publications/history/dod85.pdf`.

Vinge, Vernor. 1993. "The Coming Technological Singularity: How to Survive in the Post-Human Era." In *Vision-21: Interdisciplinary Science and Engineering in the Era of Cyberspace,* 11–22. NASA Conference Publication 10129. NASA Lewis Research Center. `http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19940022856.pdf`.

Weaver, Nik. 2013. "Paradoxes of Rational Agency and Formal Systems That Verify Their Own Soundness." Unpublished manuscript, December 21. `http://arxiv.org/abs/1312.3626`.

Yudkowsky, Eliezer. 2008. "Artificial Intelligence as a Positive and Negative Factor in Global Risk." In *Global Catastrophic Risks,* edited by Nick Bostrom and Milan M. Ćirković, 308–345. New York: Oxford University Press.

———. 2013. *The Procrastination Paradox.* Brief Technical Note. Machine Intelligence Research Institute, Berkeley, CA. `http://intelligence.org/files/ProcrastinationParadox.pdf`.

———. 2014. *Distributions Allowing Tiling of Staged Subjective EU Maximizers.* Machine Intelligence Research Institute, Berkeley, CA, May 11. Revised May 31, 2014. `http://intelligence.org/files/DistributionsAllowingTiling.pdf`.

Yudkowsky, Eliezer, and Marcello Herreshoff. 2013. *Tiling Agents for Self-Modifying AI, and the Löbian Obstacle.* Early Draft. Machine Intelligence Research Institute, Berkeley, CA. `http://intelligence.org/files/TilingAgents.pdf`.