Embedded Agents

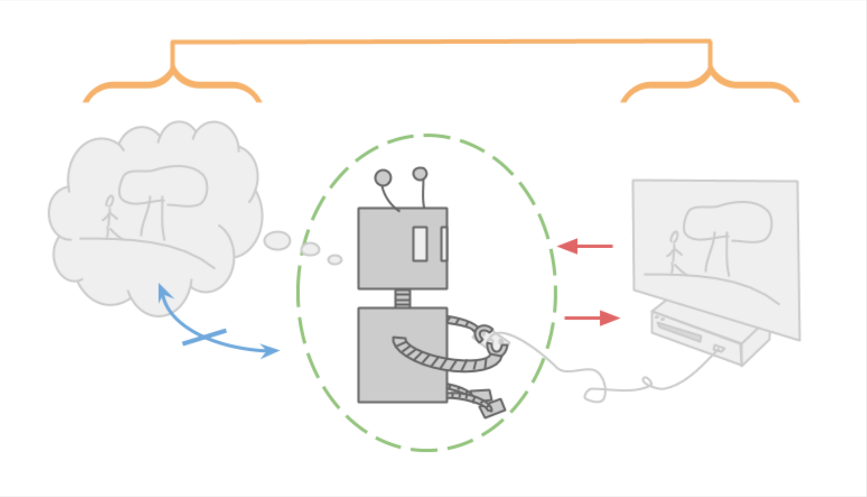
October 29, 2018 | Scott Garrabrant (https://intelligence.org/author/scott/) | Analysis (https://intelligence.org/category/analysis/)

Suppose you want to build a robot to achieve some real-world goal for you—a goal that requires the robot to learn for itself and figure out a lot of things that you don't already know.[1]

There's a complicated engineering problem here. But there's also a problem of figuring out what it even means to build a learning agent like that. What is it to optimize realistic goals in physical environments? In broad terms, how does it work?

In this series of posts, I'll point to four ways we *don't* currently know how it works, and four areas of active research aimed at figuring it out.

This is Alexei, and Alexei is playing a video game.



Like most games, this game has clear input and output channels. Alexei only observes the game through the computer screen, and only manipulates the game through the controller.

The game can be thought of as a function which takes in a sequence of button presses and outputs a sequence of pixels on the screen.

Alexei is also very smart, and capable of holding the entire video game inside his mind. If Alexei has any uncertainty, it is only over empirical facts like what game he is playing, and not over logical facts like which inputs (for a given deterministic game) will yield which outputs. This means that Alexei must also store inside his mind every possible game he could be playing.

Alexei does not, however, have to think about himself. He is only optimizing the game he is playing, and not optimizing the brain he is using to think about the game. He may still choose actions based off of value of information, but this is only to help him rule out possible games he is playing, and not to change the way in which he thinks.

In fact, Alexei can treat himself as an unchanging indivisible atom. Since he doesn't exist in the environment he's thinking about, Alexei doesn't worry about whether he'll change over time, or about any subroutines he might have to run.

Notice that all the properties I talked about are partially made possible by the fact that Alexei is cleanly separated from the environment that he is optimizing.

This is Emmy. Emmy is playing real life.

Search

intelligence.org...

Browse

All (/all-posts/)
Analysis (https://intelligence.org/category/analysis/
Conversations (https://intelligence.org/category/conversa
Guest Posts (https://intelligence.org/category/guest-posts/)
MIRI Strategy (https://intelligence.org/category/miri/)
News (https://intelligence.org/category/news/)
Newsletters (https://intelligence.org/category/newslett
Papers (https://intelligence.org/category/papers/)
Video (https://intelligence.org/category/video/)

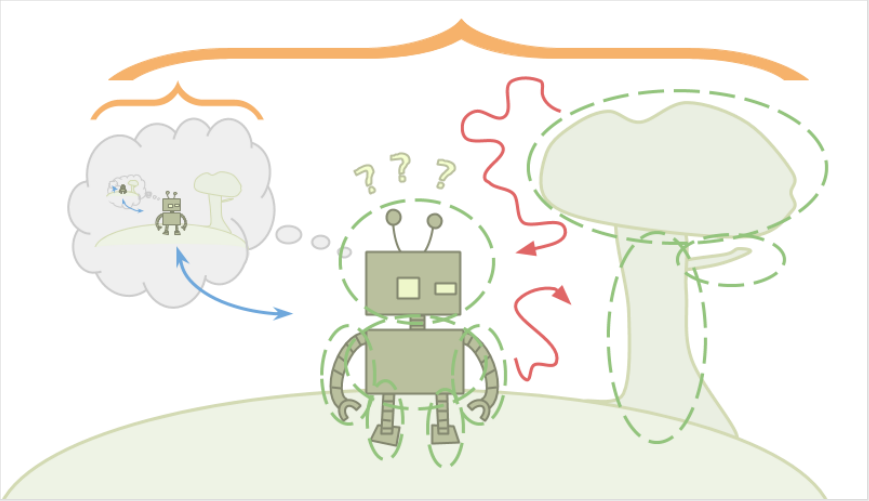
Subscribe

Join newsletter subscribers

✉ Email address

Follow @MIRIBerkeley

RSS (http://feeds.feedburner.com/miriblog)



Real life is not like a video game. The differences largely come from the fact that Emmy is within the environment that she is trying to optimize.

Alexei sees the universe as a function, and he optimizes by choosing inputs to that function that lead to greater reward than any of the other possible inputs he might choose. Emmy, on the other hand, doesn't have a function. She just has an environment, and this environment contains her.

Emmy wants to choose the best possible action, but which action Emmy chooses to take is just another fact about the environment. Emmy can reason about the part of the environment that is her decision, but since there's only one action that Emmy ends up actually taking, it's not clear what it even means for Emmy to "choose" an action that is better than the rest.

Alexei can poke the universe and see what happens. Emmy is the universe poking itself. In Emmy's case, how do we formalize the idea of "choosing" at all?

To make matters worse, since Emmy is contained within the environment, Emmy must also be smaller than the environment. This means that Emmy is incapable of storing accurate detailed models of the environment within her mind.

This causes a problem: Bayesian reasoning works by starting with a large collection of possible environments, and as you observe facts that are inconsistent with some of those environments, you rule them out. What does reasoning look like when you're not even capable of storing a single valid hypothesis for the way the world works? Emmy is going to have to use a different type of reasoning, and make updates that don't fit into the standard Bayesian framework.

Since Emmy is within the environment that she is manipulating, she is also going to be capable of self-improvement. But how can Emmy be sure that as she learns more and finds more and more ways to improve herself, she only changes herself in ways that are actually helpful? How can she be sure that she won't modify her original goals in undesirable ways?

Finally, since Emmy is contained within the environment, she can't treat herself like an atom. She is made out of the same pieces that the rest of the environment is made out of, which is what causes her to be able to think about herself.

In addition to hazards in her external environment, Emmy is going to have to worry about threats coming from within. While optimizing, Emmy might spin up other optimizers as subroutines, either intentionally or unintentionally. These subsystems can cause problems if they get too powerful and are unaligned with Emmy's goals. Emmy must figure out how to reason without spinning up intelligent subsystems, or otherwise figure out how to keep them weak, contained, or aligned fully with her goals.

Emmy is confusing, so let's go back to Alexei. Marcus Hutter's AIXI (https://arxiv.org/abs/1202.6153) framework gives a good theoretical model for how agents like Alexei work:

$$a_k \;:=\; \arg\max_{a_k} \sum_{o_k r_k} \ldots \max_{a_m} \sum_{o_m r_m} [r_k + \ldots + r_m] \sum_{q \,:\, U(q,a_1..a_m)=o_1 r_1..o_m r_m} 2^{-\ell(q)}$$

The model has an agent and an environment that interact using actions, observations, and rewards. The agent sends out an action a, and then the environment sends out both an observation o and a reward r. This process repeats at each time $k \ldots m$.

Each action is a function of all the previous action-observation-reward triples. And each observation and reward is similarly a function of these triples and the immediately preceding action.

You can imagine an agent in this framework that has full knowledge of the environment that it's interacting with. However, AIXI is used to model optimization under uncertainty about the environment. AIXI has a distribution over all possible computable environments q, and chooses actions that lead to a high expected reward under this distribution. Since it also cares about future reward, this may lead to exploring for value of information.

Under some assumptions, we can show that AIXI does reasonably well in all computable environments, in spite of its uncertainty. However, while the environments that AIXI is interacting with are computable, AIXI itself is uncomputable. The agent is made out of a different sort of stuff, a more powerful sort of stuff, than the environment.

We will call agents like AIXI and Alexei "dualistic." They exist outside of their environment, with only set interactions between agent-stuff and environment-stuff. They require the agent to be larger than the environment, and don't tend to model self-referential reasoning, because the agent is made of different stuff than what the agent reasons about.

AIXI is not alone. These dualistic assumptions show up all over our current best theories of rational agency.

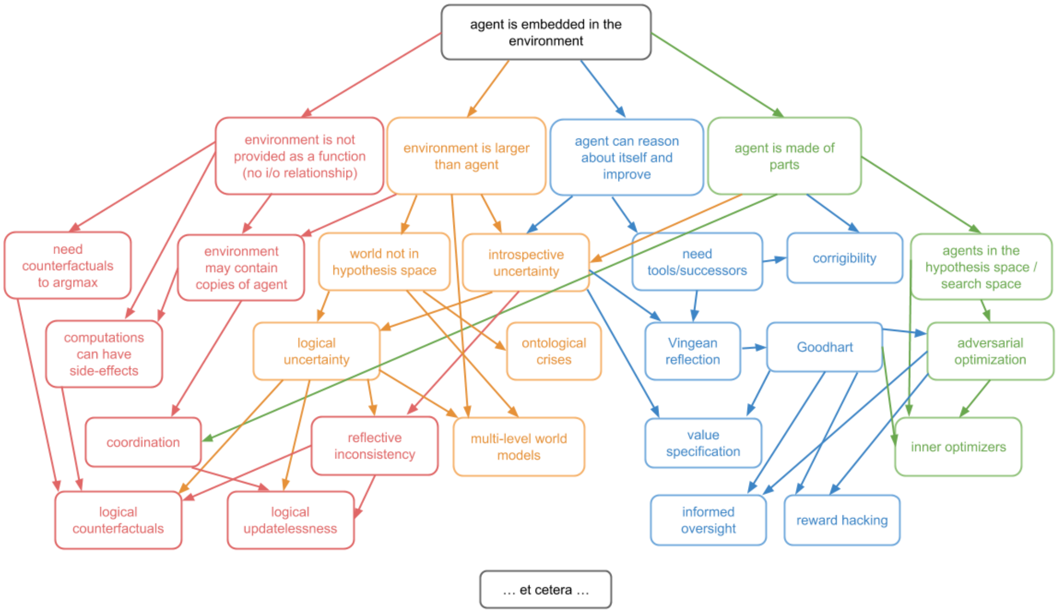
I set up AIXI as a bit of a foil, but AIXI can also be used as inspiration. When I look at AIXI, I feel like I really understand how Alexei works. This is the kind of understanding that I want to also have for Emmy.

Unfortunately, Emmy is confusing. When I talk about wanting to have a theory of "embedded agency," I mean I want to be able to understand theoretically how agents like Emmy work. That is, agents that are embedded within their environment and thus:

- do not have well-defined i/o channels;
- are smaller than their environment;
- are able to reason about themselves and self-improve;
- and are made of parts similar to the environment.

You shouldn't think of these four complications as a partition. They are very entangled with each other.

For example, the reason the agent is able to self-improve is because it is made of parts. And any time the environment is sufficiently larger than the agent, it might contain other copies of the agent, and thus destroy any well-defined i/o channels.


agent is embedded in the environment
environment is not provided as a function (no i/o relationship)
environment is larger than agent
agent can reason about itself and improve
agent is made of parts
need counterfactuals to argmax
environment may contain copies of agent
world not in hypothesis space
introspective uncertainty
need tools/successors
corrigibility
agents in the hypothesis space / search space
computations can have side-effects
logical uncertainty
ontological crises
Vingean reflection
Goodhart
adversarial optimization
coordination
reflective inconsistency
multi-level world models
value specification
inner optimizers
logical counterfactuals
logical updatelessness
informed oversight
reward hacking
… et cetera …

(https://intelligence.org/wp-content/uploads/2018/10/Embedded-Subproblems.png)

However, I will use these four complications to inspire a split of the topic of embedded agency into four subproblems. These are: decision theory, embedded world-models, robust delegation, and subsystem alignment.

Decision theory is all about embedded optimization.

The simplest model of dualistic optimization is argmax. argmax takes in a function from actions to rewards, and returns the action which leads to the highest reward under this function. Most optimization can be thought of as some variant on this. You have some space; you have a function from this space to some score, like a reward or utility; and you want to choose an input that scores highly under this function.

But we just said that a large part of what it means to be an embedded agent is that you don't have a functional environment. So now what do we do? Optimization is clearly an important part of agency, but we can't currently say what it is even in theory without making major type errors.

Some major open problems in decision theory include:

- **logical counterfactuals**: how do you reason about what *would* happen if you take action B, given that you can *prove* that you will instead take action A?
- environments that include multiple **copies of the agent**, or trustworthy predictions of the agent.
- **logical updatelessness**, which is about how to combine the very nice but very *Bayesian* world of Wei Dai's updateless decision theory (https://wiki.lesswrong.com/wiki/Updateless_decision_theory), with the much less Bayesian world of logical uncertainty.

Embedded world-models is about how you can make good models of the world that are able to fit within an agent that is much smaller than the world.

This has proven to be very difficult—first, because it means that the true universe is not in your hypothesis space, which ruins a lot of theoretical guarantees; and second, because it means we're going to have to make non-Bayesian updates as we learn, which *also* ruins a bunch of theoretical guarantees.

It is also about how to make world-models from the point of view of an observer on the inside, and resulting problems such as anthropics. Some major open problems in embedded world-models include:

- **logical uncertainty (https://intelligence.org/files/QuestionsLogicalUncertainty.pdf)**, which is about how to combine the world of logic with the world of probability.
- **multi-level modeling**, which is about how to have multiple models of the same world at different levels of description, and transition nicely between them.
- **ontological crises (https://intelligence.org/files/OntologicalCrises.pdf)**, which is what to do when you realize that your model, or even your goal, was specified using a different ontology than the real world.

Robust delegation is all about a special type of principal-agent problem. You have an initial agent that wants to make a more intelligent successor agent to help it optimize its goals. The initial agent has all of the power, because it gets to decide exactly what successor agent to make. But in another sense, the successor agent has all of the power, because it is much, much more intelligent.

From the point of view of the initial agent, the question is about creating a successor that will robustly not use its intelligence against you. From the point of view of the successor agent, the question is about, "How do you robustly learn or respect the goals of something that is stupid, manipulable, and not even using the right ontology?"

There are extra problems coming from the *Löbian obstacle* making it impossible to consistently trust things that are more powerful than you.

You can think about these problems in the context of an agent that's just learning over time, or in the context of an agent making a significant self-improvement, or in the context of an agent that's just trying to make a powerful tool.

The major open problems in robust delegation include:

- **Vingean reflection (https://intelligence.org/files/VingeanReflection.pdf)**, which is about how to reason about and trust agents that are much smarter than you, in spite of the Löbian obstacle to trust.
- **value learning (https://intelligence.org/files/ValueLearningProblem.pdf)**, which is how the successor agent can learn the goals of the initial agent in spite of that agent's stupidity and inconsistencies.
- **corrigibility (https://intelligence.org/files/Corrigibility.pdf)**, which is about how an initial agent can get a successor agent to allow (or even help with) modifications, in spite of an instrumental incentive not to.

Subsystem alignment is about how to be *one unified agent* that doesn't have subsystems that are fighting against either you or each other.

When an agent has a goal, like "saving the world," it might end up spending a large amount of its time thinking about a subgoal, like "making money." If the agent spins up a sub-agent that is only trying to make money, there are now two agents that have different goals, and this leads to a conflict. The sub-agent might suggest plans that look like they *only* make money, but actually destroy the world in order to make even more money.

The problem is: you don't just have to worry about sub-agents that you intentionally spin up. You also have to worry about spinning up sub-agents by accident. Any time you perform a search or an optimization over a sufficiently rich space that's able to contain agents, you have to worry about the space itself doing optimization. This optimization may not be exactly in line with the optimization the outer system was trying to do, but it *will* have an instrumental incentive to *look* like it's aligned.

A lot of optimization in practice uses this kind of passing the buck. You don't just find a solution; you find a thing that is able to itself search for a solution.

In theory, I don't understand how to do *optimization* at all—other than methods that look like finding a bunch of stuff that I don't understand, and seeing if it accomplishes my goal. But this is exactly the kind of thing that's *most* prone to spinning up adversarial subsystems.

The big open problem in subsystem alignment is about how to have a base-level optimizer that doesn't spin up adversarial optimizers. You can break this problem up further by considering cases where the resultant optimizers are either **intentional** or **unintentional**, and considering restricted subclasses of optimization, like **induction**.

But remember: decision theory, embedded world-models, robust delegation, and subsystem alignment are not four separate problems. They're all different subproblems of the same unified concept that is *embedded agency*.

Part 2 of this post will be coming in the next couple of days: **Decision Theory (https://intelligence.org/2018/10/31/embedded-decisions/)**.

---

1. This is part 1 of the Embedded Agency (https://intelligence.org/embedded-agency) series, by Abram Demski and Scott Garrabrant. ↩

Did you like this post? You may enjoy our other Analysis (https://intelligence.org/category/analysis/) posts, including:

- Powerful planners, not sentient software (https://intelligence.org/2015/08/18/powerful-planners-not-sentient-software/)
- Once again, a reporter thinks our positions are the opposite of what they are (https://intelligence.org/2012/11/26/once-again-a-reporter-thinks-our-positions-are-the-opposite-of-what-they-are/)
- Robust Delegation (https://intelligence.org/2018/11/04/embedded-delegation/)
- A central AI alignment problem: capabilities generalization, and the sharp left turn (https://intelligence.org/2022/07/04/a-central-ai-alignment-problem/)
- ... and many more (https://intelligence.org/category/analysis/).

Tweet

ALSO ON **MACHINE INTELLIGENCE RESEARCH INSTITUTE**

Focus on the places where you feel …

9 months ago

Crossposted from LessWrong. Writing down something I've found …

More Christiano, Cotra, and Yudkowsky on …

2 years ago

This post is a transcript of a discussion between Paul Christiano, Ajeya Cotra, …

Pausing AI Developments Isn't …

7 months ago

(Published in TIME on March 29.)   An open letter published today calls for …

The basic reasons I expect AGI ruin

6 months ago

I've been citing AGI Ruin: A List of Lethalities to explain why the situation with AI …

0 Comments

1 **Login** ▾

G

Start the discussion…

LOG IN WITH

OR SIGN UP WITH DISQUS ?

Name

♡ 1 **Share**

Best **Newest** **Oldest**

Be the first to comment.

Subscribe **Privacy** **Do Not Sell My Data**