Ω

AI Alignment Writing Day 2018

Choosing to Choose?

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

This is related to wireheading, utility functions, taste, and rationality. It is a series of puzzles meant to draw attention to certain tensions in notions of rationality and utility functions for embedded agents. I often skip the particulars of embedded situations. I often show multiple sides of an argument without staking out a clear positions myself. Even though I often write a utility function as if it were unique, this is shorthand---it is of course unique up to affine transformation.

Imagine you are in a situation with three outcomes: A, B, and C. You prefer A to C and C to B, and your preference ordering is transitive. Furthermore, assume that your preferences can be represented by a utility function.

I now give you a choice. You can either choose C, or you can choose to make a second decision between A and B. Obviously if this were the only thing going on, you would choose to choose between A and B and then you would choose A. However, there is something else going on. You know that if you chose to reject C and make the second choice then an Angel will reorder your preferences so that your new preference ordering is $B > C > A$. Thus, at the second choice, you would presumably choose B. Remember, this was the worst outcome in your original preference ordering.

Now, my question is whether a rational agent should choose to pick C, or whether it should choose to pick to choose between A and B?

Position 1: You should pick C.

An agent is rational if it makes choices that maximize its expected utility. This utility is relative to a utility function (which is grounded in preferences). Thus, what is rational for one agent may be irrational for another if they have different utility functions.

When you are in the decision problem as described above, you are trying you maximize your utility as generated by your preferences. If you choose C, you will definitely get C. If you choose to choose, then you will definitely get B, which is the worst outcome, relative to your utility function. Thus, being rational, you should choose to not choose, and take C.

Consider the following option. I give you two options: I can either murder everyone you love and give you both a pill that makes you no longer care about them and \$100 dollars, or I can give you \$10. Assume your utility is linear with money. Then, if you are willing to choose to choose in the earlier case, you have no problems with choosing to have your utility function change, and you should be fine with having everyone you love murdered. You would never actually do this. Thus, you should choose C.

Position 2: You should choose to choose.

As a rational agent, you should maximize **your** utility. Even if your utility function changes, it is still yours. If you choose C you get a middle ranked option, whereas if you choose to choose then you will choose B, which at that time will be the best outcome for you.

Consider this in the case of choosing a snack. Say there are three options: a piece of chocolate, an apple, and a banana. Say you prefer apples to chocolate, and chocolate to bananas. Now, I give you a choice: you can either have a chocolate, or I can let you take a pill which changes your preferences to preferring bananas to chocolate, and chocolate to apples. Then I let you choose between an apple and a banana.

Naturally, you would prefer to take the pill and then choose the banana. This is the same as the abstract case above, thus you should choose to choose. QED.

My own intuition from the example still tells me to pick C. I wonder if the people who would choose to choose are confusing levels of utility functions.

This is apparent when considering the example given by the person holding position 2. They use the example to try to illustrate how one's utility function changes. I take a pill which changes my preferences. I'd reply by saying that you didn't change my utility function, you just changed my taste function (so to speak).

How would this work? Well, presumably I prefer (in a simple case) one snack to another because of the flavour of the snacks. My taste buds respond in a certain way to the chemical compounds in the snacks, and then secrete such and such chemicals into my brain. It is really these chemicals (or whatever subsequent process they trigger or of which they are a part) that I desire. Thus, the pill doesn't change my utility function, but rather my taste function.

How do we represent this? The arguments to my utility function could be outputs from other functions. Imagine that all I care about is how good the food I am eating tastes, and how good the book I am reading is. Let my taste function be $\mathrm{Taste}(x)$ where x is the food I am eating, and my book function be $\mathrm{Book}(x)$ where x is the book I am reading. Then, my utility function has two inputs, y and z --- $U(y, z)$--- where $\mathrm{Taste}(x) = y$ and $\mathrm{Book}(x) = z$. If I care equally about books and food and there is some kind of comparability of the utilities, then it might be the case that $U(y, z) = U(\mathrm{Taste}(x), \mathrm{Book}(x)) = y + z = \mathrm{Taste}(x) + \mathrm{Book}(x)$.

Now, if $\mathrm{Taste}(x)$ changes, has my utility function changed?

I don't think so. My utility function is still $U(y, z) = y + z$, even if what y ends up being in different cases is different.

I can see this being a problem, though. What is to stop us from doing the following. Let $U_t(x)$ be my **true** utility function and let $U_p(y)$ be my so called *practical* utility function. Furthermore, let $U_p(y) = x$ so that $U_t(x) = U_t(U_p(y))$. If we agree that changing the taste function doesn't alter the utility function, then changing $U_p(y)$ shouldn't alter my utility function --- but this is all it is based on!

Is what matters that it is **my** utility function? If there is some other agent in the world, Atticus, and he has a utility function, I don't directly care about maximizing Atticus' utility function (I might indirectly care about trying to maximize his utility function if he is my friend or something, and we could incorporate that into my practical utility function).

Atticus cares, ultimately, only about his, and I mine. If someone where to actually change my utility function in a strong way (say, an angel) would I still be I? Or would Whispermute *qua* rational agent have gone out of existence?

This seems to be getting into difficult questions about identity and whatnot, which is an area away from which I had hoped to stay. Alack, I must venture forth.

If choosing to choose places me in a position in which I know I will have my utility function messed around with, and I think that having my utility function changed ends me *qua* rational agent, then in some sense it seems that I would cease being I were I to have my utility function changed. If this were the case, I think I would almost certainly be irrational were I to choose to choose.

Maybe for some clarity let us use a thought experiment. Suppose Maltrion is an assasin, and he serves his Master. Maltrion's utility function takes as direct input his Master's utility function. $U_{mal}(x) = x = U_{mas}(y)$.

However, Maltrion is clever, and he has found a certain spell in his Master's cabinet that he can use to change his own utility function. He contemplates changing it to be maximized by his immediate death, which he could easily fulfill due to his technical assassin skills. But he knows that this would greatly displease his master.

Given that $U_{mal}(x) = x = U_{mas}(y)$, what should he do?

Position 1: Maltrion should not use the spell. This is easily seen --- if $U_{mal}(y)$ is lowered by his action, then so is $U_{mal}(x)$ which is what is guiding Maltrion's actions. This, Maltrion should quite obviously not use the spell, as this would decrease his utility.

Position 2: Maltrion should use the spell. Maltrion wants to increase his own utility, and there is nothing essential that $U_mal(x) = x = U_mas(y)$. He is in a position to change it so that $U_{mal}(x)$ has a domain of $\{Dead, \neg Dead\}$, where $U_{mal}(Dead)$ is the highest U_{mal}

could ever be *in any possible form of U_{mal}* (suppose $U_{mal}(\text{Dead}) = \infty$). Maltrion, being a rational agent, is perfectly able to reason this through. Thus, since he wants maximize U_{mal} he should use the spell.

The Intentional Agency Experiment

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

We would like to discern the intentions of a hyperintelligent, possibly malicious agent which has every incentive to conceal its evil intentions from us. But what even is intention? What does it mean for an agent to work towards a goal?

Consider the lowly ant and the immobile rock. Intuitively, we feel one has (some) agency and the other doesn't, while a human has more agency than either of them. Yet, a sceptic might object that ants seek out sugar and rocks fall down but that there is no intrinsic difference between the goal of eating yummy sweets and minimising gravitational energy.

Intention is a property that an agent has with respect to a goal. Intention is not a binary value, a number or even a topological vector space. Rather, it is a certain constellation of counterfactuals.

***

Let W be a world, which we imagine as a causal model in the sense of Pearl: a directed acyclic graph with nodes and attached random variables $N_0, \ldots, N_k$. Let R be an agent. We imagine R to be a little robot - so **not** a hyperintelligent malignant AI- and we'd like to test whether it has a goal G, say $G = (N_0 = 10)$. To do so we are going to run an Intentional Agency Experiment: we ask R to choose an action A_1 from its possible actions $B = \{a_1, \ldots, a_n\}$.

Out of the possible actions B one a_{best} is the 'best' action for R if it has goal $G = (N_0 = 10)$ in the sense that $P(G|do(a_{best})) \geq P(G|do(a_i))$ for $i = 1, \ldots, n$

If R doesn't choose $A_1 = do(a_{best})$, great! We're done; R doesn't have goal G. If R *does* choose $A_1 = do(a_{best})$, we provide it with a new piece of (counterfactual) information $P(G|A_1) = 0$ and offer the option of changing its action. From the remaining actions there is one next best actions a_{nbest}. Given the information $P(G|A_1) = 0$ if R does not choose $A_2 = do(a_{nbest})$ we stop, if R does we provide it with the information $P(G|A_2) = 0$ and continue as before.

At each round we assign more and more agency to R. Rather, than a binary 'Yes, R has agency' or 'No, R has no agency' we imagine a continuum going from a rock, which has no possible actions, to an ant, which might pass some of the tests but not all, to humans and beyond.

***

Q: What if R isn't acting rational? What if it doesn't know all the details of W? What if it knows more? What if R has bounded computation? What if...

A: The above is merely a simple model that tries to capture intent; one can complicate it as needed. Most of these objections come down to the possible inability of R to choose the best action (given infinite compute and full knowledge of W). To remedy this we might allow the Intentional Agency Experiment to continue if R chooses an action that is close to optimal but not optimal. We may introduce a Time to Think parameter when we consider bounded computational agents, etc. Once again, the point is not to assign a binary value of goal intention to an agent, rather it is to assign it a degree of agency.

Q: What if R knows that we are testing and tries to deceive us?

A: Yes, this breaks the model.

Q: Counterfactuals are weird and might not exist. Even if they did, the Intentional Agency Experiment is impossible to execute in practise.

Despite the Intentional Agency Experiment being an idealisation, we may approximate it in the real world. For instance, if we'd like to test the intention of an ant to seek out a sugar source (as opposed to a random walk) we might first check if it moves towards the sugar source; if it does we block off this route towards the sugar source and sees whether it tries to circumvent it. In fact, it could be argued that this is the way we test agency in real life.

Two agents can have the same source code and optimise different utility functions

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

I believe that it is possible for two agents to have the exact same source code while (in some sense) optimising two different utility functions.

I take a utility function to be a function from possible world states to real numbers. When I say that an agent is optimising a utility function I mean something like that the agent is "pushing" its environment towards states with higher values according to said utility function. This concept is not entirely unambiguous, but I don't think its necessary to try to make it more explicit here. By source code I mean the same thing as everyone else means by source code.

Now, consider an agent which has a goal like "gain resources" (in some intuitive sense). Say that two copies of this agent are placed in a shared environment. These agents will now push the environment towards different states, and are therefore (under the definition I gave above) optimising different utility functions.

Conditioning, Counterfactuals, Exploration, and Gears

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

The view of counterfactuals as just conditioning on low-probability events has a lot going for it. To begin with, in a bayesian setting, updates are done by conditioning. A probability distribution, conditioned on some event x (an imaginary update), and a probability distribution after actually seeing x (an actual update) will be identical.

There is an issue with conditioning on low-probability events, however. When x has a low probability, the conditional probability $\frac{P(x \wedge y)}{P(x)}$ has division by a small number, which amplifies noise and small changes in the probability of the conjunction, so estimates of probability conditional on lower-probability events are more unstable. The worst-case version of this is conditioning on a zero-probability event, because the probability distribution after conditioning can be literally *anything* without affecting the original probability distribution. One useful intution for this is that probabilities conditional on x are going to be less accurate, when you've seen very few instances of x occuring, as the sample size is too small to draw strong conclusions.

However, in the logical inductor setting, it is possible to get around this with infinite exploration in the limit. If you act unpredictably enough to take bad actions with some (very small) probability, then in the limit, you'll experiment enough with bad actions to have well-defined conditional probabilities on taking actions you have (a limiting) probability 0 of taking. The counterfactuals of standard conditioning are those where the exploration step occured, just as the counterfactuals of modal UDT are those where the agents implicit chicken step went off because it found a spurious proof in a nonstandard model of PA.

Now, this notion of counterfactuals can have bad effects, because zooming in on the little slice of probability mass where you do x is different from the intuitive notion of counterfacting on doing x. Counterfactual on me walking off a cliff, I'd be badly hurt, but conditional on me doing so I'd probably also have some sort of brain lesion. Similar problems exist with Troll Bridge, and this mechanism is the reason why logical inductors converge to not giving Omega money in a version of Newcomb's problem where Omega can't predict the exploration step. Conditional on them 2-boxing, they are probably exploring in an unpredictable way, which catches Omega unaware and earns more money.

However, there's no better notion of counterfactuals that currently exists, and in fully general environments, this is probably as well as you can do. In multi-armed bandit problems, there are many actions with unknown payoff, and the agent must converge to figuring out the best one. Pretty much all multi-armed bandit algorithms involve experimenting with actions that are worse than baseline, which is a pretty strong clue that exploration into bad outcomes is necessary for good performance in arbitrary environments. If you're in a world that will reward or punish you in arbitrary if-then fashion for selecting any action, then learning the reward given by three of the actions doesn't help you figure out the reward of the fourth action. Also, in a similar spirit as Troll Bridge, if there's a lever that shocks you, but only when you pull it in the spirit of experimentation, then if you don't have access to exactly how the lever is working, but just the external behavior, it's perfectly reasonable to believe that it just always shocks you (after all, it's done that all other times it was tried).

And yet, despite these arguments, humans can make successful engineering designs operating in realms they don't have personal experience with. And humans don't seem to reason about what-ifs by checking what they think about the probability of x and $x \wedge y$, and comparing the two. Even when thinking about stuff with medium-high probability, humans seem to reason by imagining some world where the thing is true, and then reasoning about consequences of the thing. To put it another way, humans are using some notion of $\text{counterfactuals}_{\text{human}}$ in place of conditional probabilities.

Why can humans do this at all?

Well, physics has the nice property that if you know some sort of initial state, then you can make accurate predictions about what will happen as a result. And these laws have proven their durability under a bunch of strange circumstances that don't typically occur in nature. Put another way, in the multi-armed bandit case, knowing the output of three levers doesn't tell you what the fourth will do, while physics has far more correlation among the various levers/interventions on the environment, so it makes sense to trust the predicted output of pulling a lever you've never pulled before. Understanding how the environment responds to one sequence of actions tells you quite a bit about how things would go if you took some different sequence of actions. (Also, as a side note, conditioning-based counterfactuals work very badly with full trees of actions in sequence, due to combinatorial explosion and the resulting decrease in the probability of any particular action sequence)

The environment of math, and figuring out which algorithms you control when you take some action you don't, appears to be intermediate between the case of fully general multi-armed bandit problems, and physics, though I'm unsure of this.

Now, to take a detour to Abram's old post on gears . I'll exerpt a specific part.

> Here, I'm siding with David Deutsch's account in the first chapter of *The Fabric of Reality*. He argues that understanding and predictive capability are distinct, and that understanding is about having good explanations. I may not accept his whole critique of Bayesianism, but that much of his view seems right to me. Unfortunately, he doesn't give a *technical* account of what "explanation" and "understanding" could be.

Well, if you already have maxed-out predictive capability, what extra thing does understanding buy you? What useful thing is captured by "understanding" that isn't captured by "predictive capability"?

I'd personally put it this way. Predictive capability is how accurate you are about what will happen in the environment. But when you truly understand something, you can use that to figure out actions and interventions to get the environment to exhibit weird behavior that wouldn't have precedent in the past sequence of events. You "understand" something when you have a compact set of rules and constraints telling you how a change in starting conditions affects some set of other conditions and properties, which feels connected to the notion of a gears-level model.

To summarize, conditioning-counterfactuals are very likely the most general type, but when the environment (whether it be physics or math) has the property that the change induced by a different starting condition is describable by a much smaller program than an if-then table for all starting conditions, then it makes sense to call to call it a "legitimate counterfactual". The notion of there being something beyond epsilon-exploration is closely tied to having compact descriptions of the environment and how it behaves under interventions, instead of the max-entropy prior where you can't say anything confidently about what happens when you take a different action than you normally do, and this also seems closely tied to Abram's notion of a "gears-level model"

There are interesting paralells to this in the AIXI setting. The "models" would be the Turing machines that may be the environment, and the Turing machines are set up such that any action sequence could be input into them and they would behave in some predictable way. This attains the property of accurately predicting consequences of various action sequences AIXI doesn't take if the world it is interacting with is low-complexity, for much the same reason as humans can reason through consequences of situations they have never encountered using rules that accurately describe the situations they have encountered.

However if AIXI has some high-probability world (according to the starting distribution) where an action is very dangerous, it will avoid that action, at least until it can rule out that world by some other means. As Leike and Hutter entertainingly show, this "Bayesian paranoia" can make AIXI behave arbitrarily badly, just by choosing the universal turing machine appropriately, to assign high probability to a world where AIXI goes to hell and gets 0 reward forever if it ever takes some action.

This actually seems acceptable to me. Just don't be born with a bad prior. Or, at least, there may be some self-fulfilling prophecies, but it's better then having exploration into bad outcomes in every world with irreversible traps. In particular, note that exploration steps are reflectively inconsistent, because AIXI (when considering the future) will do worse (according to the current probability distribution over Turing machines) if it uses exploration, rather than using the current probability distribution. AIXI is optimal according to the environment distribution it starts with, while AIXI with exploration is not.

Probability is fake, frequency is real

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

Consider the Sleeping Beauty problem. What do we mean by fair coin? It is meant that the coin will have 50-50 probability of heads or tails. But that is fake. It will ether come up heads or tail, because the real world is deterministic. It is true that I don't know the outcome. I don't know if I am in a world of type "the coin will come up heads" or a world of type "the coin will come up tails". But in this situation I should be allowed to put what ever prior I want on the coins behavior.

Consider the Born rule of quantum mechanics. If I measure the spin of an electron, then I will entangle the large apparatus that is the my measuring equipment with the spin of the electron. We say that there are now two Everett branches, one where the apparatus measured spin up and one where the apparatus measured spin down. Before I read of the result, I don't know which Hilbert branch I am in. I could be in ether, and I should be allowed to have what ever prior I want. So why the Born rule? Why to I do I believe that the square amplitude is **the** correct way of assigning probability to which Hilbert branch I am in?

I believe in the Born rule because of the frequency of experimental outcomes in the past. The distribution of galaxies in the sky can be traced back to the Born rule. I don't have the gears on what is causing the Born rule, but there are something undeniably real about galaxies that trumps mere philosophical Bayesian arguments about freedom of priors.

Imagine that you are offered a bet. Should you take it or not? There are several argument about what you should do in different situations. For example, if you have finite amount of money, you should maximize the E(log(money)) for each bet, (see e.g. Kelly criterion). However, every such argument I have ever seen, is assuming that you will be confronted by a large number of similar bets. This is because probabilities only relay make sense if you sample enough times from the random distribution you are considering.

The notion of "fair coin" does not make sense if the coin is flipped only once. The right way to view the Sleeping Beauty problem is to view it in it in the context of Repeated Sleeping Beauty.

Next: Repeated (and improved) Sleeping Beauty problem

Repeated (and improved) Sleeping Beauty problem

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

Follow up to: Probability is fake, frequency is real

There is something wrong with the normal formulation of the Sleeping Beauty problem. More precisely, there is something wrong about postulating a single "fair" random coin flip. So here is an improved version of the Sleeping Beauty problem. After explaining the setup, I will recover the normal Sleeping Beauty problem, but in a more well defined way.

There are no truly random coins. There are only pseudo random coins which has the property that you don't have the capacity to calculate the outcome. A **fair** pseudo random coin have the additional property that when flipped enough times, the ratio of *Heads* v.s. *Tails* will approach one. Note that fairness is only defined if you actually flip the coin a sufficient number of times. Because of this, the Sleeping Beauty problem should be a repeated game.

(Alternatively, you could solve this by using counterfactuals. However, we don't yet know how to deal with counterfactuals. Also, I suspect that any method of handling counterfactuals will be, at best, useful but wrong.)

Repeated Sleeping Beauty setup: Every Sunday a mysterious person flips a pseudo random fair coin. If the coin comes up *Heads*, Sleeping Beauty will wake up on Monday, and then sleep for the rest of the week. If the coin comes up *Tails*, she will wake up on Monday and Tuesday and then sleep for the rest of the week. No-one is telling Sleeping Beauty what is going on, she gets to rely on her own past experiences.

Every morning when Sleeping Beauty wakes up she does not know what day it is. However there is an easy experiment she can do to find out, namely asking anyone she meets on the street. Because Sleeping is a curious person, she is keeping a science journal. Every day she finds out what day it is and writes it down. She soon notices some patterns.

1) There are two kinds of days, Monday and Tuesday.

2) Every Tuesday is followed by a Monday.

3) A Monday can be followed by either a Tuesday or a Monday.

After some more time she starts to notice the frequencies of which different days occur.

1/3 of days are Mondays that are followed by Monday (corresponds to *Heads* & Monday)

1/3 of days are Mondays that are followed by Tuesday (corresponds to *Tails* & Monday)

1/3 of days are Tuesdays (corresponds to *Tails* & Tuesday)

Sleeping Beauty tries to find more patterns in the data, but none of the more complicated hypothesizes she can come up with survives further observation.

Recovering the original Sleeping Beauty problem: Sleeping Beauty have been slacking off for a few days, and not asking for what day it was. What likelihood should she assign to the current day being a Monday followed by Monday?

The obvious answer based on Sleeping Beauty's own experience is 1/3.

Conclusion and after-though: If you take your probability from how things have played out in the past, you will learn the Thirder position / Self-indication assumption (SIA). Also, doing what has worked well in the past leads to Evidential decision theory (EDT). This is a sad fact of the universe, because EDT combined with SIA leads to a sort of double counting of actions which add up to the wrong policy [citation].

Logical Uncertainty and Functional Decision Theory

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

(I used this paper as a reference for functional decision theory (FDT), which is essentially an improved version of timeless decision theory and updateless decision theory)

This post is a reflection on decision processes that refer to themselves (call them introspective agents if you like) with a view toward investigating the problem of counterfactuals. It ruminates on the fact that logical uncertainty about the output of a decision process is often a good thing for agents.

Let's begin with a tentative definition. A deterministic agent B is *autonomous with respect to* A if A cannot predict B. A has logical uncertainty about B's actions. This occurs, for instance, in Newcomb's problem, when the output of the predictor depends upon what A thinks. We can extend this definition to stochastic worlds and say that B is autonomous with respect to A if A is logically uncertain about the probability distribution of B's actions (check this---it might not be the right definition). Somewhat obviously, agents can be autonomous with respect to each other (humans), mutually transparent (two trivial agents) or there can be asymmetry (the Newcomb problem). An agent can also be *autonomous with respect to itself,* such as we are. I want to argue here that this is a good and necessary thing.

The absence of the last possibility, self-autonomy or self-transcendence, seems to be the source of problems relating to counterfactuals (the "five-and-ten" problem). Self-transparent agents, given models of the world as Turing machines, are "fixed" about their own behavior in a strange and rigid way. They embody "fixed points". Sometimes this is good---it means that they are consistent. But if an agent equipped with a logical inductor believes that it will, given a choice between a $5 and a $10 bill, always take the $5, will in fact do so self-consistently. This is insane and rigid. On the other hand, it would also be consistent if the logical inductor believed that the agent would always take the $10. Updatelessness is tricky; reflective stability gives fixed points but not automatically the right ones.

On the other hand, FDT works because it considers each of the "fixed points" (consistent possible successor-worlds) separately, and chooses the one with greatest expected utility. Can we make a logical inductor do this? We can construct all possible logical inductors (different traders etc) or sample the space in hope of finding all the fixed points, but a thorough fix (haha) is not within my sight. Fixed points, as always and unto the ages of ages, are a pain in the butt.

An FDT agent is autonomous with respect to herself. She does not know the outcome of her decision process until she has completed it. There is logical uncertainty about her decision which cannot be removed without rendering her vulnerable to psychopathic pest control workers and similarly foul characters. This very uncertainty about herself has a protective effect. How absurd it is to know what one will choose before having chosen!

It has been proposed to remedy this issue by introducing the "ϵ-chicken rule": if A believes that it will make some choice (take the $5) with greater than $1 - \epsilon$ credence, then it must not make that choice. In this post, Scott Garrabrant showed that this approach solves the 5 and 10, but introduces other problems. Maybe it is the wrong kind of uncertainty? It is dumb, blind randomness, absolutely different from an FDT agent's thoughtful and reflective logical uncertainty about herself.

The situation with reflective oracles seems to be similar. They are reflectively self-consistent, but the fixed points they represent are not guaranteed to be good. It would be best to generate the reflective oracle that gives the Pareto optimum, but this takes extra work. *Making a choice*, picking an oracle, deciding between possible worlds, reduces logical uncertainty and is therefore a kind of `update' of self-knowledge.

All the above is mere intuition with some funny language, but it points to the positive and protective role of logical uncertainty in decision-making. So take the above as a nudge toward a line of research which I think could yield a rich harvest.

A framework for thinking about wireheading

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

[Epistemic status: Writing down in words something that isn't very complicated, but is still good to have written down.]

A great deal of ink has been spilled over the possibility of a sufficiently intelligent AI noticing that whatever it has been positively reinforced for seems to be *very* strongly correlated with the floating point value stored in its memory labelled “utility function”, and so through some unauthorized mechanism editing this value and defending it from being edited back in some manner hostile to humans. I'll reason here by analogy with humans, while agreeing that they might not be the best example.

“Headwires” are (given a little thought) not difficult to obtain for humans -- heroin is freely available on the black market, and most humans know that, when delivered into the bloodstream, it generates “reward signal”. Yet most have no desire to try it. Why is this?

Asking any human, they will answer something along the lines of ”becoming addicted to heroin will not help me achieve my goals” (or some proxy for this: spending all your money and becoming homeless is not very helpful in achieving one's goals for most values of “goals”.) Whatever the effects of heroin, the actual pain and pleasure that the human brains have experienced has led us to become optimizers of very different things, which a state of such poverty is not helpful for.

Reasoning analogously to AI, we would hope that, to avoid this, a superhuman AI trained by some kind of reinforcement learning has the following properties:

1. While being trained on “human values” (good luck with that!) the AI must not be allowed to hack its own utility function.
2. Whatever local optima the training process that generated the AI ends up in (perhaps reinforcement learning of some kind) assigns some probability to the AI optimising what we care about.
3. (most importantly) The AI realizes that trying wireheading will lead it to become an AI which prefers wireheading over aim it currently has, which would be detrimental to this aim.

I think this is an important enough issue that some empirical testing might be needed to shed some light. Item (3) seems to be the most difficult to implement; we in the real world have the benefit of observing the effects of hard drugs on their unfortunate victims and avoiding them ourselves, so a multi-agent environment in which our AI realizes it is in the same situation as other agents looks like a first outline of a way forward here.

Bayesian Probability is for things that are Space-like Separated from You

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

First, I should explain what I mean by space-like separated from you. Imagine a world that looks like a Bayesian network, and imagine that you are a node in that Bayesian network. If there is a path from you to another node following edges in the network, I will say that node is time-like separated from you, and in your future. If there is a path from another node to you, I will say that node is time-like separated from you, and in your past. Otherwise, I will say that the node is space-like separated from you.

Nodes in your past can be thought of as things that you observe. When you think about physics, it sure does seem like there are a lot of things in your past that you do not observe, but I am not thinking about physics-time, I am thinking about logical-time. If something is in your past, but has no effect on what algorithm you are running on what observations you get, then it might as well be considered as space-like separated from you. If you compute how everything in the universe evaluates, the space-like separated things are the things that can be evaluated either before or after you, since their output does not change yours or vice-versa. If you partially observe a fact, then I want to say you can decompose that fact into the part that you observed and the part that you didn't, and say that the part you observed is in your past, while the part you didn't observe is space-like separated from you. (Whether or not you actually can decompose things like this is complicated, and related to whether or not you can use the tickle defense is the smoking lesion problem.)

Nodes in your future can be thought of as things that you control. These are not always things that you want to control. For example, you control the output of "You assign probability less than 1/2 to this sentence," but perhaps you wish you didn't. Again, if you partially control a fact, I want to say that (maybe) you can break that fact into multiple nodes, some of which you control, and some of which you don't.

So, you know the things in your past, so there is no need for probability there. You don't know the things in your future, or things that are space-like separated from you. (Maybe. I'm not sure that talking about knowing things you control is not just a type error.) You may have cached that you should use Bayesian probability to deal with things you are uncertain about. You may have this justified by the fact that if you don't use Bayesian probability, there is a Pareto improvement that will cause you to predict better in all worlds. The problem is that the standard justifications of Bayesian probability are in a framework where the facts that you are uncertain about are not in any way affected by whether or not you believe them! Therefore, our reasons for liking Bayesian probability do not apply to our uncertainty about the things that are in our future! Note that many things in our future (like our future observations) are also in the future of things that are space-like separated from us, so we want to use Bayes to reason about those things in order to have better beliefs about our observations.

I claim that logical inductors do not feel entirely Bayesian, and this might be why. They can't if they are able to think about sentences like "You assign probability less than 1/2 to this sentence."

A universal score for optimizers

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

[Epistemic status: the idea is simple and intuitive, so I wouldn't be surprised if it has appeared before. If so, I would appreciate pointers.]

Eliezer defines optimization power as the ability to produce solutions higher on the preference ordering. This feels right, but it would be nice to convert this intuition into a score that can be measured, a universal metric that allows one to meaningfully state that AlphaZero is better at Go than Deep Blue was at chess. In this post I try accomplishing just that. I start by giving a candidate metric and some of its properties. I then discuss how it can be approximately computed in complex settings, and describe a somewhat paradoxical implication of adopting this definition for repeated games.

For simplicity I will assume a deterministic, finite-action-space environment (extending everything to MDPs with uncountable actions doesn't seem to be a fundamental obstacle). The action space is A, the outcome space is X, utility of an agent under question is $u(x(a))$. Because $x(a)$ is deterministic I will just write $u(a)$ for brevity. Note that we can represent sequential decision problems in this framework (e.g. Sudoku), elements of A would then be vectors of individual actions. We write $n(S)$ for the size of set S.

Definition. Suppose we observe an agent with utility function $u(a)$ take an action $\hat{a}$, then we say this agent has a C-score of:

$$C(u, \hat{a}) = -\log \frac{n(a \mid u(a) \geq u(\hat{a}))}{n(A)}$$

Intuitively, C-score is inversely proportional to the fraction of outcomes that are as good for the agent as the one it managed to obtain. One interpretation of this formula is that the agent is competing against a bot that just picks a random action (random-bot), and then $-C(u, \hat{a})$ is log probability of losing in this competition.

Some properties of C(u,a):

- C-score is independent of computational power employed
- It's impossible to compute C-score without knowing the utility function
- Improving from being in top 1% outcomes to top 0.1% is as "hard" as improving from 10% to 1%
- One can use Monte-Carlo to generate lower bounds of the form "w.h.p. C-score of this agent is at least Y"
- Strategy of taking a random action produces the same score in any setting (under some assumptions it might also be approximately true for the strategy of trying k random moves and picking the best one)

Approximating C-score

Can we estimate C-score for complex environments and agents? For example, suppose we have an agent playing chess against a known distribution of opponents (or, more simply, against random-bot), and utility function is the probability of winning the game (action space A then is a set of policies, i.e. mappings from state to action). Can we measure C-score of this agent without using an unrealistic amount of compute?

Clearly, the naive Monte-Carlo approach I mentioned above is a no-go in this scenario: the probability that a randomly sampled action would be in the set $n(a|u(a) \geq u(\hat{a}))$ is tiny (if the agent is any good), and we will never sample this set.

I have a couple of potential ideas here:

- In case of having access to a better optimizer than the one being measured, one can use rare event sampling, that is bias the sampling procedure towards good policies and then account for this bias when computing the C-score. This approach nicely fits with the intuition "you need to be at least about as smart as the agent to measure how smart it is"
- If the game is DP/MDP and the agent employs value function estimation for picking moves, one can try looking at how much the agent improves when enhanced with MCTS, and then try inferring C-score from it. I don't have good explanations for why and how this can work, only an intuition that this is a meaningful approach to try.

Repeated game paradox

I'll use a very simple game to illustrate this issue. An agent picks a number between 1 and 10 and utility of the agent equals to the number chosen. This game is repeated T times, so agent's total utility is the sum of numbers it returned in all T games.

If T=1 and the agent picks 8, we have $C_1 = -\log(3/10)$ (there are 3/10 actions that yield utility of at least 8)

If T=2 and the agent picks 8 twice, we get $C_2 = -\log(15/100)$

$C_3 = -\log(84/1000)$, $C_4 = -\log(495/10^4)$

Thus an agent that finds a good solution once at t=1, and then repeats the same action until the end, obtains higher and higher scores for bigger T. This feels like an artifact of the definition (coming from how volumes grow with dimensions), rather than the agent being a genuinely better optimizer. Is there a score formula that has similar properties but doesn't suffer from this paradox?

Thanks to Linda, Joar and Chris for helpful discussions that led to this post.

An environment for studying counterfactuals

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

Summary: I introduce a decision theory framework where successful agents are those with good counterfactuals.

Motivation

The problem of logical counterfactuals is how to define probabilities $P(\phi|A() = a)$ when $A() = a$ is known to be false. (I'll ignore more general counterfactuals $P(\phi|\psi)$ in this post.)

The theory of logical induction provides a joint distribution over sentences, so the problem becomes: How do you condition on $A() = a$ when $A() = a$ has negligible probability?

Exploration tries to solve this by making sure that $A() = a$ never has negligible probability. But it doesn't work in problems like Agent Simulates Predictor that contain predictors who can't tell when the agent explores.

A better solution is early exploration, which uses an early stage of the logical inductor to do exploration. But then the later stages of the inductor know that $A() = a$ is false, and we're back where we started.

I'm going to describe an environment that captures these features of the problem — it's got reflection, early exploration, counterfactuals, and a Bayesian update that stands in for the evolution of a logical inductor.

Informal definition

The agent outputs counterfactual distributions $p(U = u|A = a)$. This determines an expected utility for each action. Most of the time, an action is chosen for the agent that maximizes this expected utility. But a small fraction of the time, an exploration action is chosen instead.

The agent receives an observation O as input, from which it can infer whether exploration will occur. The agent also receives a prior $\mathbb{P}$ as input, and this prior accurately reflects the behavior of the agent as a function of O and $\mathbb{P}$. (This uses a fixed-point theorem.)

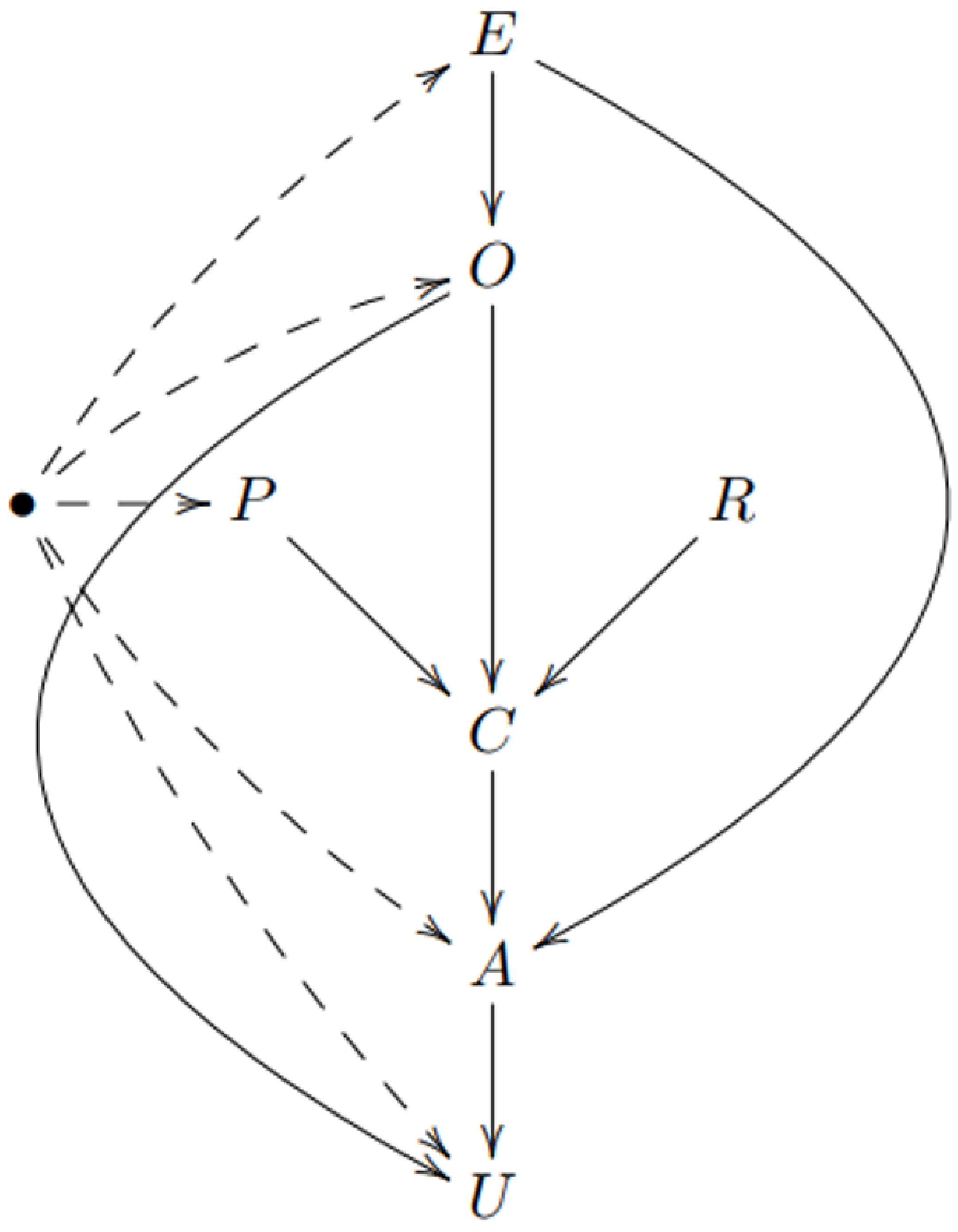
If action a is chosen, then the counterfactual $p(U = u|A = a)$ is factual; the rest are counterfactual. We judge an agent according to how accurate its factual counterfactual is, in addition to how much utility it gets.

Here's an agent that does okay in this environment: It adopts P as its epistemic state and ignores O. Because of exploration, it can compute counterfactuals by conditioning. This agent does okay but not great, since it ignores O.

You could try to make a better agent as follows: Adopt P as a prior and then do a Bayesian update on O. But now you've inferred whether exploration occurs, so some actions have probability zero, and it's not clear how to compute counterfactuals.

If you find a good agent for this environment, you'll probably have learned something about making good counterfactuals.

Formal definition



A decision problem consists of a tuple of random variables:

- A is a finite set of actions.
- E = A ∪ {∗} determines whether the agent explores. If E = ∗, no exploration takes place.
- O is a finite set of observations.

- $U = \mathbb{R}$ is the space of utilities.
- $\mathcal{P} = \Delta(E \times O \times A \times U)$ is the space of distributions over the above variables.
- $R = \{0,1\}^\infty$ represents an infinite source of random bits that the agent can use.
- $C = \Delta(U)^A$ is the agent's output, representing a counterfactual distribution $P(U = u | A = a)$ for each action.

along with some likelihoods:

- $P(E = a) = \frac{\varepsilon}{|A|}$ for all $a \in A$, and $P(E = *) = 1 - \varepsilon$, for some choice of ε.
- Likelihoods $P(O = o | E = e)$, depending on the problem.
- i.i.d. uniform distributions on each bit of R.
- A is mostly determined by E and C as follows: If $E = *$, then $A = \text{argmax}_a \, \mathbb{E}_{C(a)}[U]$. (If there is a tie, A is undetermined.) Otherwise, $A = E$.
- A distribution over U conditional on each value of O and A, depending on the problem.

An agent is a function $\mathcal{P} \times O \times R \to C$. A decision and an agent together *almost* determine a joint distribution over all the variables. What's missing is $\mathcal{P}$ and tiebreakers for A. These are determined by finding a fixed point satisfying:

- If P is the resulting marginal distribution over $E \times O \times A \times U$, then $\mathcal{P} = P$.
- For each $o \in O$ and $r \in R$, there is a distribution over the set $\text{argmax}_a \, \mathbb{E}_{C(a)}[U]$ such that A is sampled from that distribution.

(I might prove the existence of a fixed point in a comment.)

We'll informally say that an agent does well on a decision problem if, for every fixed point, the following are true:

- $\mathbb{E}[U]$ is high.
- The factual counterfactual is accurate — say, it's close to the marginal over U conditional on the true action in total variation distance: $\mathbb{E}[\,\delta(C(a), P(-|O = o, A = a)) \,|\, O = o, A = a]$.

Future work

I have an idea for defining an optimal agent for every decision problem in this family; I'll explore that in another post.

Once we find a general solution, we'd ideally transfer it to the setting of logical induction, and then we'd have logical counterfactuals.

Mechanistic Transparency for Machine Learning

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

Cross-posted on my blog.

[EDIT (added Jan 2023): it's come to my attention that this post was likely influenced by conversations I had with Chris Olah related to the distinction between standard interpretability and the type called "mechanistic", as well as early experiments he had which became the 'circuits' sequence of papers - my sincere apologies for not making this clearer earlier.]

Lately I've been trying to come up with a thread of AI alignment research that (a) I can concretely see how it significantly contributes to actually building aligned AI and (b) seems like something that I could actually make progress on. After some thinking and narrowing down possibilities, I've come up with one -- basically, a particular angle on machine learning transparency research.

The angle that I'm interested in is what I'll call *mechanistic* transparency. This roughly means developing tools that take a neural network designed to do well on some task, and outputting something like pseudocode for what algorithm the neural network implements that could be read and understood by developers of AI systems, without having to actually run the system. This pseudocode might use high-level primitives like 'sort' or 'argmax' or 'detect cats', that should themselves be able to be reduced to pseudocode of a similar type, until eventually it is ideally reduced to a very small part of the original neural network, small enough that one could understand its functional behaviour with pen and paper within an hour. These tools might also slightly modify the network to make it more amenable to this analysis in such a way that the modified network performs approximately as well as the original network.

There are a few properties that this pseudocode must satisfy. Firstly, it must be faithful to the network that is explained, such that if one substitutes in the pseudocode for each high-level primitive recursively, the result should be the original neural network, or a network close enough to the original that the differences are irrelevant (although just in case, the reconstructed network that is exactly explained should presumably be the one deployed). Secondly, the high-level primitives must be somewhat understandable: the pseudocode for a 256-layer neural network for image classification should not be `output = f2(f1(input))` where `f1` is the action of the first 128 layers and `f2` is the action of the next 128 layers, but rather break down into edge detectors being used to find floppy ears and spheres and textures, and those being combined in reasonable ways to form judgements of what the image depicts. The high-level primitives should be as human-understandable as possible, ideally 'carving the computation at the joints' by representing any independent sub-computations or repeated applications of the same function (so, for instance, if a convolutional network is represented as if it were fully connected, these tools should be able to recover convolutional structure). Finally, the high-level primitives in the pseudocode should ideally be understandable enough to be modularised and used in different places for the same function.

This agenda nicely relates to some existing work in machine learning. For instance, I think that there are strong synergies with research on compression of neural networks. This is partially due to background models about compression being related to understanding (see the ideas in common between Kolmogorov complexity, MDL, Solomonoff induction, and Martin-Löf randomness), and partially due to object-level details about this research. For example, sparsification seems related to increased modularity, which should make it easier to write understandable pseudocode. Another example is the efficacy of weight quantisation, which means that the least significant bits of the weights aren't very important, indicating that the relations between the high-level primitives should be modular in an understandable way and not have crucial details depend on some of the least significant bits of the output.

The Distill post on the building blocks of interpretability includes some other examples of work that I feel is relevant. For instance, work on using matrix factorisation to group neurons seems very related to constructing high-level primitives, and work on neuron visualisation should help with understanding the high-level primitives if their output corresponds to a subset of neurons in the original network.

I'm excited about this agenda because I see it as giving the developers of AI systems tools to detect and correct properties of their AI systems that they see as undesirable, without having to deploy the system in a test environment that they must laboriously ensure is adequately sandboxed. You could imagine developers checking if their systems conform to theories of aligned AI, or detecting any 'deceive the human' subroutine that might exist. I see this as fairly robustly useful, being helpful in most stories of how one would build an aligned AI. The exception is if AGI is built without things which look like modern machine learning algorithms, which I see as unlikely, and at any rate hope that lessons transfer to the methods which are used.

I also believe that this line of research has a shot at working for systems which act in the world. It seems hard for me to describe how I detect laptops given visual informations, but given visual primitives like 'there's a laptop there', it seems much easier for me to describe how I play tetris or even go. As such, I would expect tools developed in this way to illuminate the strategy followed by tetris-playing DQNs by referring to high-level primitives like 'locate T tetronimo', that themselves would have to be understood using neuron visualisation techniques.

Visual primitives are probably not the only things that would be hard to fully understand using the pseudocode technique. In cases where humans evade oversight by other humans, I assert that it is often not due to consequentialist reasoning, but rather due to avoiding things which are frustrating or irritating, where frustration/irritation is hard to introspect on but seems to reliably steer away from oversight in cases where that oversight would be negative. A possible reason that this frustration/irritation is hard to introspect upon is that it is complicated and hard to decompose cleanly, like our object recognition systems are. Similarly, you could imagine that one high-level primitive that guides the AI system's behaviour is hard to decompose and needs techniques like neuron visualisation to understand. However, at least the mechanistic decomposition allowed us to locate this subsystem and determine how it is used in the network, guiding the tests we perform on it. Furthermore, in the case of humans, it's quite possible that our frustration/irritation is hard to introspect upon not because it's hard to understand, but rather because it's strategically better to not be able to introspect upon it (see the ideas in the book The Elephant in the Brain), suggesting that this problem might be less severe than it seems.

Bounding Goodhart's Law

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

Goodhart's law seems to suggest that errors in utility or reward function specification are necessarily bad in sense that an optimal policy for the incorrect reward function would result in low return according to the true reward. But how strong is this effect?

Suppose the reward function were only slightly wrong. Can the resulting policy be arbitrarily bad according to the true reward or is it only slightly worse? It turns out the answer is "only slightly worse" (for the appropriate definition of "slightly wrong").

Definitions

Consider a Markov Decision Process (MDP) $M = (S, A, T, R^*)$ where

- S is the set of states,
- A is the set of actions,
- $T : S \times A \times S \to \mathbb{R}$ are the conditional transition probabilities, and
- $R^* : S \times A \to \mathbb{R}$ is the reward function. (Note: "reward" is standard terminology for MDPs but it's fine to think of this as "utility")

A policy $\pi : S \times A \to \mathbb{R}$ is a mapping from states to distributions over actions with$\pi(s, a) = \Pr(a|s)$.

Any given policy π induces a distribution $\sigma(\pi)$ over states in this MDP. If we are concerned about average reward we can take $\sigma(\pi)$ to be the stationary distribution or, if the environment is episodic, we can take $\sigma(\pi)$ to be the distribution of states visited during the episode. The exact definition is not particularly important for us.

Define the return of policy π according to reward function R to be

$$G(\pi, R) = E_\pi[R(s, a)] = \sum_{s \in S} \sum_{a \in A} \sigma_s(\pi)\pi(s, a)R(s, a)$$

Goodhart Regret

Suppose we have an approximate reward signal $\hat{R}$ and we use it to specify a policy $\hat{\pi}$. How bad is $\hat{\pi}$ according to the true reward R^*?

More specifically, what is the regret of using $\hat{\pi}$ compared to the optimal policy π^*? Formally,

$$\text{Regret}(\hat{\pi}) = G(\pi^*, R^*) - G(\hat{\pi}, R^*)$$

We can expand this as

$$G(\pi^*, R^*) - G(\hat{\pi}, R^*)$$

$$= [G(\pi^*, R^*) - G(\pi^*, \hat{R})] + [G(\pi^*, \hat{R}) - G(\hat{\pi}, \hat{R})] + [G(\hat{\pi}, \hat{R}) - G(\hat{\pi}, R^*)]$$

Let $\epsilon \geq 0$, then $\text{Regret}(\hat{\pi}) \leq 3\epsilon$ if the following conditions are satisfied by $\hat{R}$ and $\hat{\pi}$:

1. $G(\pi^*, R^*) - G(\pi^*, \hat{R}) \leq \epsilon$

2. $G(\pi^*, \hat{R}) - G(\hat{\pi}, \hat{R}) \leq \epsilon$

3. $G(\hat{\pi}, \hat{R}) - G(\hat{\pi}, R^*) \leq \epsilon$

Condition 2 says that $\hat{\pi}$ is not much worse than π^* when measured against $\hat{R}$. That is what we expect if we designed $\hat{\pi}$ to be specifically good at $\hat{R}$, so condition 2 is just a formalization of the notion that $\hat{\pi}$ is tailored to $\hat{R}$.

Conditions 1 and 3 compare a fixed policy against two different reward functions. In general for policy π and reward functions R and R',

$$G(\pi, R) - G(\pi, R') = E_\pi[R(s,a) - R'(s,a)] \leq \max_{s,a}(R(s,a) - R'(s,a))$$

Result: Uniformly Bounded Error

Assume that we have a reward approximation $\hat{R}$ with uniformly bounded error. That is, $\forall s \in S, \forall a \in A, |R^*(s,a) - \hat{R}(s,a)| < \epsilon$. Take $\hat{\pi} = \text{argmax}_\pi G(\pi, \hat{R})$.

Then $\text{Regret}(\hat{\pi}) < 2\epsilon$. (Condition 2 has bound 0 in this case).

Result: One-sided Error Bounds

A uniform bound on the error is a stronger condition than we really need. The conditions on $\hat{R}$ can be re-written:

1. $E_{\pi^*}[R^*(s,a) - \hat{R}(s,a)] \leq \epsilon$; $\hat{R}$ does not substantially underestimate the reward in the regions of state-space that are frequently visited by π^*.

3. $E_{\hat{\pi}}[R^*(s,a) - \hat{R}(s,a)] \geq -\epsilon$; $\hat{R}$ does not substantially overestimate the reward in the regions of state-space that are frequently visited by $\hat{\pi}$.

In other words, it doesn't matter if the reward estimate is too low for states that π^* doesn't want to visit anyways. This tells us that we should prefer biasing our reward approximation to be low in the absence of more information. We do need to be careful about not overestimating where $\hat{\pi}$ does visit, which is made difficult by the fact that condition 2 pushes $\hat{\pi}$ to visit states that $\hat{R}$ assigns high reward.

If we don't know what π^* is then it might be difficult to ensure we don't underestimate the reward over $\sigma(\pi^*)$. We probably have access to $\hat{\pi}$ so we might expect to have better reward approximations over $\sigma(\hat{\pi})$ and therefore have an easier time satisfying condition 3 than 1.

Non-Optimal π^*

The bound on $G(\pi^*, R^*) - G(\hat{\pi}, R^*)$ does not actually require π^* to be an optimal policy for R^*. We can take π^* to be any policy and if we can satisfy the 3 bounds then $\hat{\pi}$ will be not much worse than π^* (and could be better). Using a known policy for π^* likely makes it easier to satisfy condition 1, which is an expectation over the state-action distribution of π^*.

Example: Human Reward Learning

Since π^* need not be optimal, we can take π^* to be the policy of a human and try to learn our reward / utility function. *Note: this is a very flawed proposal, its purpose is to demonstrate how one might think about using these bounds in reward learning, not to be a concrete example of safe reward learning.*

Suppose that there is some ideal reward function R^* that we'd like to approximate. We don't know R^* in general but imagine we can evaluate it for state-action pairs that are performed by a human. Let $D_H = \{(s_i, a_i, r_i = R^*(s_i, a_i))\}_{i=1}^{N}$ be a collection of human demonstrations with labelled reward.

Consider the following algorithm:

1. Fit $\hat{R}$ to D_H so that it does not underestimate.

2. Set $\hat{\pi} = \text{argmax}_\pi G(\pi, \hat{R})$.

3. Collect a batch $\hat{D} = \{(s_j, a_j)\}_{j=1}^{M}$ of demonstrations from $\hat{\pi}$ (no reward labels).

4. Augment D_H with some additional human demonstrations.

5. Modify $\hat{R}$ to assign lower reward to all of $\hat{D}$ while not underestimating on D_H

6. Repeat from 2.

Assuming $\hat{R}$ is sufficiently expressive, this algorithm pushes $\hat{R}$ and $\hat{\pi}$ towards satisfying all three conditions: $\hat{R}$ does not underestimate on the distribution of human-visited states, $\hat{R}$ is otherwise as low as possible on states $\hat{\pi}$ visits, and $\hat{\pi}$ is optimal for $\hat{R}$. If these are all met, the resulting policy would be no worse than the human at maximizing R^* and possibly much better.

Comments and Takeaways

Goodhart's law is more impactful in the context of the sparse reward setting, in which approximation means deciding *what* to value, not how much to value. Consider a state space that is the real line and suppose that the true reward function is $R^*(s,a) = 1[s = 1]$ where $1[\cdot]$ is the indicator function.

If we estimate

$$\hat{R}_1(s, a) = 1[s = 0.999]$$

then Goodhart's law applies and we can expect that a policy optimized for $\hat{R}_1$ will have high regret on R^*.

On the other hand, if we estimate

$$\hat{R}_2(s, a) = 0.999 \cdot 1[s = 1] + 0.001 \cdot 1[s = 0.999] - 10 \cdot 1[s = 2]$$

then the regret bounds apply and a policy optimized for $\hat{R}_2$ will do very well on R^*.

- Assigning high value to the wrong things is bad.
- Assigning the wrong value to the right things is not too bad.
- Throwing a large amount of optimization power against an incorrect utility function is not always bad.

A comment on the IDA-AlphaGoZero metaphor; capabilities versus alignment

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

Iterated Distillation and Amplification has often been compared to AlphaGo Zero. But in the case of a Go-playing AI, people often don't see a distinction between alignment and capability, instead measuring success on a single axis: ability to win games of Go. But I think a useful analogy can still be drawn that separates out ways in which a Go-playing AI is aligned from ways in which it is capable.

A well-aligned but not very capable Go-playing agent would be an agent that is best modeled as trying to win at Go, as opposed to trying to optimize some other feature of sequence of board positions, but still does not win against moderately skilled opponents. A very capable but poorly aligned Go-playing agent would be an agent that is very good at causing the Go games that it plays to have certain properties other than the agent winning. One way to create such an agent would be to take a value network that rates board positions somewhat arbitrarily, and then use Monte Carlo tree search to find policies that cause the game to progress through board states that are rated highly by the value network.

To a certain extent, this could actually happen to something like AlphaGo Zero. If some bad board positions are mistakenly assigned high values, AlphaGo Zero will use Monte Carlo tree search to search for ways of getting to those board positions (and similarly for ways to avoid good board positions that are mistakenly assigned low values). But it will simultaneously be correcting the mistaken values of these board positions as it notices that its predictions of the values of future board positions that follow from them are biased. Thus the Monte Carlo tree search stage increases both capability and alignment. Meanwhile, the policy network retraining stage should be expected to decrease both capability and alignment, since it is merely approximating the results of the Monte Carlo tree search. The fact that this works shows that the extent to which Monte Carlo tree search increases alignment and capability exceeds the extent to which policy network retraining decreases them.

It seems to me that this is the model that Iterated Distillation and Amplification should be trying to emulate. The amplification stage will both increase capability by increasing the available computing power, and increase alignment because of the human overseer. The distillation stage will decrease both capability and alignment by imperfectly approximating the amplified agent. The important thing is not to drive the extent to which distillation decreases alignment down to zero (which may be impossible), but to ensure that the distillation stage does not cause alignment problems that will not be corrected in the subsequent amplification stage.

Dependent Type Theory and Zero-Shot Reasoning

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

I.

When I sit down to write code, I can safely throw everything I can think of at the problem and just keep trying to run the code until it works—and when it works, I can actually see it working. If my code runs, it means I actually had to understand enough of the moving parts to get it to run. This is especially true in languages like Haskell with strong type constraints preventing me from doing anything too stupid.

When I sit down to prove a theorem, I have to be much more careful. One misstep in my reasoning and my whole proof will be invalid, without me even knowing I made a mistake until I or someone else catches it later. If my proof goes through, it's possible I understood enough of the moving parts to make it work, but it's also possible I didn't.

The standard solution to the problem of not having something to test your proof against when doing math is to use a proof checker, which lets you turn an arbitrary math problem into a programming problem. Instead of having to just get the right answer without ever being able to actually check your answer, you get lots of chances to test possible solutions.

Proof checkers are great, but traditional proof checkers are also pretty limited. Concepts like equality require specialized machinery like paramodulation to properly resolve, introducing new data types requires writing out long lists of PA-like axioms and in general modern functional programming tools that make writing out math easier like pattern-matching, ADTs, monads, etc. are just missing. When I wrote my own theorem prover/proof checker a while ago I ran into exactly these sorts of problems.

Dependent type theory is the modern way to solve all of these problems. Both equality and the natural numbers are defined just by giving the constructors, no axioms needed. For-alls and implications are just functions. Proving theorems really just feels like writing Haskell code.

I'm not going to try and give a dependent type theory tutorial here, because I don't think I would be able to do better than the one I used, Theorem Proving in Lean, which I highly recommend taking a look at. That being said, I do want to give a couple examples of what I mean before I move on. Don't worry if you don't understand this part, I just want to show you some actual code in a dependent type theory.

I said equality and the natural numbers were just data types, but to show you what I mean here is the actual definition of the natural numbers in the Lean standard library

```
            inductive nat
| zero : nat
| succ (n : nat) : nat
```

where : means "has type of." Those few lines are all that are required to completely define the semantics of the natural numbers—the induction rule is automatically synthesized from the above definition.

Equality is slightly more complex, but only because you need a little bit more boilerplate. Here is the actual definition of equality in the Lean standard library

```
            inductive eq {α : Sort u} (a : α) : α → Prop
| refl : eq a
```

which is really just the Lean translation of the Haskell

```
            data Equals a = Refl
```

where `eq a` represents `a = a` and `refl` stands for reflexivity, the rule that `a = a` always holds.

II.

Even despite all of the advances in automated proof checking with dependent type theory, there's still a big disadvantage to typing up all of your proofs in a dependent type theory like Lean: formality. When you write a proof by hand, you have the ability to elide over all sorts of details that you have to make explicit when you type them up into a proof checker. Thus, the main reason not to use a proof checker is the very reason to use one in the first place.

I think there exists a middle ground between these two extremes, however. One thing which has most impressed me working with Nate Soares is his ability to construct massive mathematical proofs on paper without a proof checker. When I asked him how he did this, he told me he used the type checker in his head. I've been trying to do this as well recently, and I want to try to share what that looks like.

Lawvere's theorem is a general result in category theory that states that if there exists a surjective function $f : X \to X \to A$ (technically $X \to A^X$ but we'll elide that detail), then A has the property that any function $h : A \to A$ has some fixed point $a : A$ such that $ha = a$. To give you a sense of how important this theorem is, it's the general result underlying both Gödel's incompleteness theorems and Löb's theorem. I claim that Lawvere's theorem is nearly trivial to prove if you just imagine you have a type checker in your head.

Our starting environment, where the $\vdash$ represents the thing we're trying to prove, is

$$f : X \to X \to A$$

$$\mathrm{Sur}_f : \forall g : X \to A.\ \exists x : X.\ fx = g$$

$$h : A \to A$$

$$\vdash \exists a : A.\ ha = a$$

from which point the only possible way to proceed is to use Sur_f, which, under the interpretation where for-alls are functions, requires us to pass it some $g : X \to A$. If you think about how to construct a g with the right type out of what we currently have available, there really aren't very many options. In fact, I think there are basically just the two: $gx = fxx$ and $gx = h(fxx)$, but since we know we're trying to prove something about h, the definition with h in it seems like the safest bet. Thus, we get

$$g : X \to A$$
$$gx = h(fxx)$$
$$\vdash \exists a : A.\, ha = a$$

which, passing g to Sur_f, gives us

$$\text{Sur}_{f \text{ of } g} : \exists x : X.\, fx = g$$
$$\vdash \exists a : A.\, ha = a$$

which we can (classically) unpack into

$$x : X$$
$$geq : fx = g$$
$$\vdash \exists a : A.\, ha = a$$

and if two functions are equal, then they must be equal on all inputs, and the only possible input available is x, so we get

$$geq_x : fxx = gx = h(fxx)$$
$$\vdash \exists a : A.\, ha = a$$

which, where $a = fxx$, completes the proof.

III.

I was talking with Scott Garrabrant late one night recently and he gave me the following problem: how do you get a fixed number of DFA-based robots to traverse an arbitrary maze (if the robots can locally communicate with each other)? My approach to this problem was to come up with and then try to falsify various possible solutions. I started with a hypothesis, threw it against counterexamples, fixed it to resolve the counterexamples, and iterated. If I could find a hypothesis which I could prove was unfalsifiable, then I'd be done.

When Scott noticed I was using this approach, he remarked on how different it was than what he was used to when doing math. Scott's approach, instead, was to just start proving all of the things he could about the system until he managed to prove that he had a solution. Thus, while I was working backwards by coming up with possible solutions, Scott was working forwards by expanding the scope of what he knew until he found the solution.

I think that the difference between my approach and Scott's approach elucidates an important distinction regarding strategies for solving problems in general. In Alex Zhu's post on zero-shot reasoning he talked about the difference between zero-shot reasoning, where you solve the problem without ever testing your solution, and many-shot reasoning, where you solve the problem by testing your solution repeatedly. In many ways, Scott's approach to the robot problem was a zero-shot approach, whereas my approach was a many-shot approach.

One thing that's interesting in this analogy, however, is that neither Scott or I were actually using any tools to solve the problem aside from our own brains. Thus, in some sense, both of our approaches were zero-shot. And yet, my approach certainly feels far more many-shot than Scott's. I'm going to call what I did in this situation many-shot reasoning for a zero-shot problem, and I think this idea is one that applies a lot more broadly than just in that conversation. One possible way to understand the distinction between MIRI's and Paul Christiano's strategies towards AI alignment, for example, is using this distinction. In this interpretation, MIRI is solving the zero-shot problem with zero-shot reasoning, whereas Paul is solving the zero-shot problem with many-shot reasoning.

In particular, I think this idea maps well onto why I find dependent type theory so useful for solving math problems. When I write up a math problem in Lean, I'm turning what is fundamentally a zero-shot problem into something more like a many-shot problem, since it gives me a way to constantly be testing my hypotheses. But something even more interesting is happening when I'm using the type checker in my head. In that case, I'm doing something more like what I did in the robot problem, where I'm using many-shot reasoning to solve a zero-shot problem. I'm never actually using anything to test my hypotheses—instead, I'm building a model in my head of the thing which would test my hypotheses, and then testing them against that.

I think this hints at what a solution to the zero-shot reasoning problem might look like: if you can build a model of the corresponding many-shot problem which is accurate enough, then you can use many-shot reasoning on that model to solve the original zero-shot problem.

Conceptual problems with utility functions

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

[Epistemic status: strong intuitions that I have had for a long time, but not yet compared with other people's intuitions, and I am not sure how well I can convey my intuitions.]

So, a lot of agent foundations research seems to be built on the premise that agenthood is about maximizing the expected value of some utility function, or about having actions that bear some relation to some utility function, or something like that. I think that this premise is wrong except in special cases like where there is only one agent in the world. I don't know exactly how to explain this, so I will just say some stuff.

The Ultimatum Game is a standard example of a game in which there are many Pareto optimal outcomes, and no clear way to choose between them. So if we imagine two "perfectly rational" agents playing the Ultimatum Game against each other, what happens? I think that this question is meaningless. OK, so what happens in real life? People use their notions of fairness to resolve situations like the Ultimatum Game. Fairness is a part of human values, so essentially the answer is that an agent's values are used to resolve the ambiguity of multiple Pareto optimal outcomes.

But wait! In the classical understanding, a utility function is supposed to encode *all* information about an agent's values. So if there is a notion of fairness relevant to a real-life Ultimatum Game based on monetary payouts, then it supposedly means that the utility function is not just the same as the monetary payouts, and the game is not a true Ultimatum Game at all. But then what is a true Ultimatum Game? Does such a mythical beast even exist? Eliezer had to invent a fairly far-fetched scenario before he found something that he was willing to call the "true Prisoner's dilemma".

But even in the "true Prisoner's dilemma", the utility function does not appear to capture all of Eliezer's values -- he seems to still be motivated to say that "cooperate" is the right answer based on symmetry and hope, which again appear to be human values. So I propose that in fact there is no "true Prisoner's dilemma", simply because calling something a dilemma is asking you to resolve it with your own values, but your own values are allegedly encapsulated by the utility function which is subject to the dilemma.

I propose instead that agenthood is a robustness to these sorts of games, a sort of continual supply of values which are sufficient to escape from any logical argument purporting to prove that there is no determinate answer as to what action you should take. There is no "pure essence of rational agenthood" that we can simply graft onto the "utility function" of human values to make an aligned AI, because human values are not only a function on world-states to be learned but also a part of the decisionmaking process itself. This seems to suggest radically different approaches to the alignment problem than what is present so far, but I am not sure what they are yet.

No, I won't go there, it feels like you're trying to Pascal-mug me

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

The goal of this blog-post is to explore the intuition that drives the feeling that the agent who arrives at the conclusion that they should pay Pascal's mugger is being unreasonable, and whether this can be tied in with the analysis of a logical inductor as an algorithm not exploitable by an efficiently computable trader. An analysis of this intuition may help us to design the decision theory of a future Friendly AI so that it is not vulnerable to being Pascal-mugged, which is plausibly a desirable goal from the point of view of obtaining behaviour in alignment with our values.

So, then, consider the following conversation that I recently had with another person at a MIRI workshop.

"Do you know Amanda Askell's work? She takes Pascal's wager seriously. If there really are outcomes in the outcome space with infinite utility and non-infinitesimal probability, then that deserves your attention."

"Oh, but that's very Goodhartable."

A thought in close proximity to this would be that that line of reasoning would open you up to exploitation by agents not aligned with your core values. This is plausibly the main driver behind the intuition that paying Pascal's mugger is unreasonable, a decision theory like that makes you too exploitable.

The work done by MIRI on the analysis of the notion of logical induction identifies the key desirable property of a logical inductor algorithm as producing a pattern of belief states evolving over time which isn't exploitable by any efficiently computable trader.

Traders studying the history of your belief states won't be able to identify any pattern that they can exploit with a polynomial-time computable algorithm, so when such patterns emerge in the sentences you have already proved, you'll make the necessary "market correction" in your own belief states so that such patterns don't make you exploitable. Thus, being a good logical inductor is related to having patterns of behaviour that are not exploitable. We see here some hint of a connection with the notion that being Pascal's-mugging-resilient is related to not being exploitable.

Plausibly a useful research project would be to explore whether some similar formalisable notion in an AI's decision theory could capture the essence of the thought behind "Oh, Pascal's-mugging reasoning looks too suspicious". Perhaps the desired line of reasoning might be "A regular habit of being persuaded by Pascal-mugging style reasoning would make me vulnerable to exploitation by agents not aligned with my core values, independently of whether such exploitation is in fact likely to occur given my beliefs about the universe. If I can see that an argument for a particular conclusion about which action to take manifests that kind of vulnerability, I ought to invest more computing time before actually taking the action". It is not hard to imagine that natural selection might have been a driver for such a heuristic and this may explain why Pascal-mugging-style reasoning "feels suspicious" to humans even when they haven't yet constructed the decision-theoretic argument that yields the conclusion not to pay the mugger. If we can build a similar heuristic into an AI's decision theory, this may help to filter out "unintended interpretations" of what kind of behaviour would optimise what humans care about, such as large-scale wireheading, or investing all your computing resources into figuring out some way to hack the laws of physics to produce infinite utility despite very low probability of a positive pay-off.

We also see here some hint of a connection between the insights that drove the "correct solution" to what a logical inductor should be and the "correct solution" to what the best decision theory should be. Exploring whether this is in some way generalisable to other domains may also be a line of thought that deserves further examination.

Conditions under which misaligned subagents can (not) arise in classifiers

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

Core claim: *Misaligned subagents are very unlikely to arise in a classification algorithm unless that algorithm is directly or indirectly (e.g. in a subtask) modeling interactions through time at a significant level of complexity.*

Definition 1: Agent - a function from inputs and internal state (or memory) to an output / action and new internal state. Note that this includes things that would not usually be considered as "agents" - e.g. plants or bacteria. Also note that not all "agents" of this type have consistent (or even coherent) "goals"

This definition of agent might be considered too broad; the reason I have decided to use it is that I believe it covers basically everything that could be dangerous - if an AI is not an agent under this definition, then I think it is extremely likely that this AI would be safe.

Definition 2: A function that was selected by an optimization procedure has a misaligned subagent if it spawns a subprocess that is an agent whose "goals" are different from (and potentially in conflict with) the optimization criteria.

Example: Consider an optimization process that selects for functions that accurately predict human actions, and assume that this optimization process finds a function that does this prediction by creating an extremely accurate simulations of humans. These simulations would be misaligned subagents, since humans are agents and the goals of the simulations would likely be very different from "predict human actions accurately".

For brevity, let us abbreviate classifiers with misaligned subagents as **CWMS**. Note that I might use classifier a bit more broadly than the strict definition - for example, I may call certain more general question-answering machines "classifiers". I do not believe this directly affects the general argument.

Claim 1: An agent will "perform" "better" than a memoryless function given the same sequence of inputs only if (almost) every input is highly correlated with the previous input. To phrase this in a different way, having a "memory" only helps if your "memory" gives good evidence for either what is going to happen next or what to do next.

Claim / Assumption 2: For most optimization processes powerful enough to find a classifier that performs well, we have:

$$P\text{ (Optimization process finds a CWMS)} \approx \frac{\text{Density of CWMS that "perform well"}}{\text{Density of all classifiers that "perform well"}}$$

There might be certain optimization processes that lean more towards CWMS (or toward classifiers without misaligned subagents), but I think this is a reasonable base assumption given our current level of information.

Claim 3: For a certain task, if most good classifiers for that task are CWMS, then there exists one or more subtasks that involve processing highly correlated inputs, and doing some of these subtasks very well is important for being a good classifier. This follows from Claims 1 and 2.

Conversely, if such key subtasks do not exist, most good classifiers will not be CWMS

Intuition 4: For most tasks which have key subtasks of the type mentioned in Claim 3, those subtasks are very likely to involve some sort of modeling how things change / interact through time (example: understanding the contents of a video or the meaning of a paragraph in a book require this type of modeling).

Intuition 5: There are a lot of interesting, difficult tasks where modeling how things change through time is not key for solving the task.

Claim / Intuition 6: Optimizers for the tasks mentioned in Intuition 5 have very low probability of finding a CWMS. This follows from Claims 2 and 3 and Intuition 4.

Thanks to Scott Garrabrant and Evan Hubinger for discussing some the ideas in this post with me.

Complete Class: Consequentialist Foundations

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

The fundamentals of Bayesian thinking have been justified in many ways over the years. Most people here have heard of the VNM axioms and Dutch Book arguments. Far fewer, I think, have heard of the Complete Class Theorems (CCT).

Here, I explain why I think of CCT as a more purely consequentialist foundation for decision theory. I also show how complete-class style arguments play a role is social choice theory, justifying utilitarianism and a version of futarchy. This means CCT acts as a bridging analogy between single-agent decisions and collective decisions, therefore shedding some light on how a pile of agent-like pieces can come together and act like one agent. To me, this suggests a potentially rich vein of intellectual ore.

I have some ideas about modifying CCT to be more interesting for MIRI-style decision theory, but I'll only do a little of that here, mostly gesturing at the problems with CCT which could motivate such modifications.

Background

My Motives

This post is a continuation of what I started in Generalizing Foundations of Decision Theory and Generalizing Foundations of Decision Theory II. The core motivation is to understand the justification for existing decision theory very well, see which assumptions are weakest, and see what happens when we remove them.

There is also a secondary motivation in human (ir)rationality: to the extent foundational arguments are *real reasons* why rational behavior is better than irrational behavior, one might expect these arguments to be helpful in teaching or training rationality. This is related to my criterion of consequentialism: the argument in favor of Bayesian decision theory should directly point to *why it matters*.

With respect to this second quest, CCT is interesting because Dutch Book and money-pump arguments point out irrationality in agents by *exploiting* the irrational agent. CCT is more amenable to a model in which you point out irrationality by *helping* the irrational agent. I am working on a more thorough expansion of that view with some co-authors.

Other Foundations

(Skip this section if you just want to know about CCT, and not why I claim it is better than alternatives.)

I give an overview of many proposed foundational arguments for Bayesianism in the first post in this series. I called out Dutch Book and money-pump arguments as the most promising, in terms of motivating decision theory only from "winning". The second post in the series attempted to motivate all of decision theory from only those two arguments (extending work of Stuart Armstrong along those lines), and succeeded. However, the resulting argument was in itself not very satisfying. If you look at the structure of the argument, it justifies constraints on decisions via problems which would occur in hypothetical games involving money. Many philosophers have argued that the Dutch Book argument is in fact a way of illustrating inconsistency in belief, rather than truly an argument that you must be consistent or else. I think this is right. I now think this is a serious flaw behind both Dutch Book and money-pump arguments. There is no pure consequentialist reason to constrain decisions based on consistency relationships with thought experiments.

The position I'm defending in the current post has much in common with the paper Actualist Rationality by C. Manski. My disagreement with him lies in his dismissal of CCT as yet another bad argument. In my view, CCT seems to address his concerns almost precisely!

Caveat --

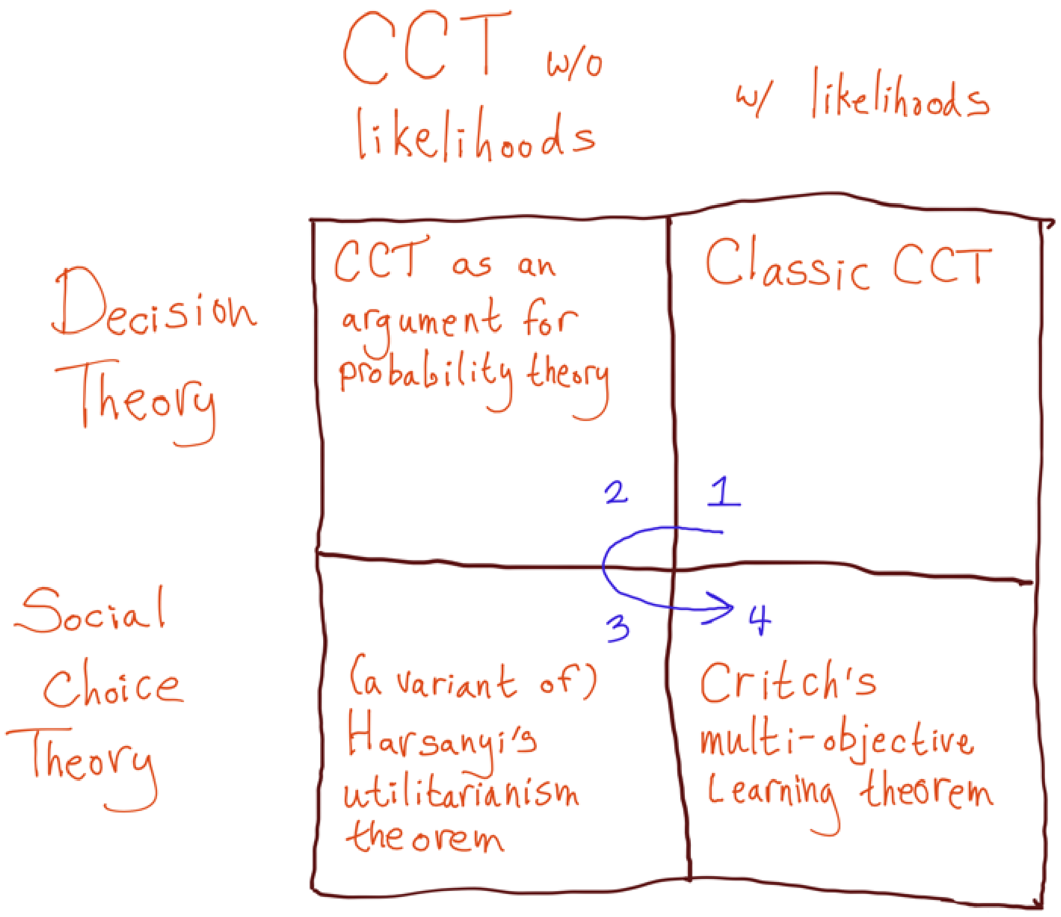
Dutch Book arguments are *fairly* practical. Betting with people, or asking them to consider hypothetical bets, is a useful tool. It may even be what convinces someone to use probabilities to represent degrees of belief. However, the argument falls apart if you examine it too closely, or at least requires extra assumptions which you have to argue in a different way. Simply put, belief is not literally the same thing as willingness to bet. Consequentialist decision theories are in the business of relating beliefs to actions, not relating beliefs to betting behavior.

Similarly, money-pump arguments can sometimes be extremely practical. The resource you're pumped of doesn't need to be money -- it can simply be the cost of thinking longer. If you spin forever between different options because you prefer strawberry ice cream to chocolate and chocolate to vanilla and vanilla to strawberry, you will not get any ice cream. However, the set-up to money pump *assumes* that you will not notice this happening; whatever the extra cost of indecision is, it is placed outside of the considerations which can influence your decision.

So, Dutch Book "defines" belief as willingness-to-bet, and money-pump "defines" preference as willingness-to-pay; in doing so, both arguments put the justification of decision theory into hypothetical exploitation scenarios which are not quite the same as the actual decisions we face. If these were the best justifications for consequentialism we could muster, I would be somewhat dissatisfied, but would likely leave it alone. Fortunately, a better alternative exists: complete class theorems.

Four Complete Class Theorems

For a thorough introduction to complete class theorems, I recommend Peter Hoff's course notes. I'm going to walk through four complete class theorems dealing with what I think are particularly interesting cases. Here's a map:



CCT w/o likelihoods
w/ likelihoods
Decision Theory
Social Choice Theory
CCT as an argument for probability theory
Classic CCT
(a variant of) Harsanyi's utilitarianism theorem
Critch's multi-objective Learning theorem
2
1
3
4

In words: first we'll look at the standard setup, which assumes likelihood functions. Then we will remove the assumption of likelihood functions, since we want to argue for probability theory from scratch. Then, we will switch from talking about decision theory to social choice theory, and use CCT to derive a variant of Harsanyi's utilitarian theorem, AKA Harsanyi's social aggregation theorem, which tells us about cooperation between agents with common beliefs (but different utility functions). Finally, we'll add likelihoods back in. This gets us a version of Critch's multi-objective learning framework, which tells us about cooperation between agents with different beliefs *and* different utility functions.

I think of *Harsanyi's utilitarianism theorem* as the best justification for utilitarianism, in much the same way that I think of CCT as the best justification for Bayesian decision theory. It is not an argument that *your personal values* are necessarily utilitarian-altruism. However, it *is* a strong argument for utilitarian altruism as the most coherent way to care about others; and furthermore, to the extent that groups can make rational decisions, I think it is an extremely strong argument that the group decision should be utilitarian. AlexMennen discusses the theorem and implications for CEV here.

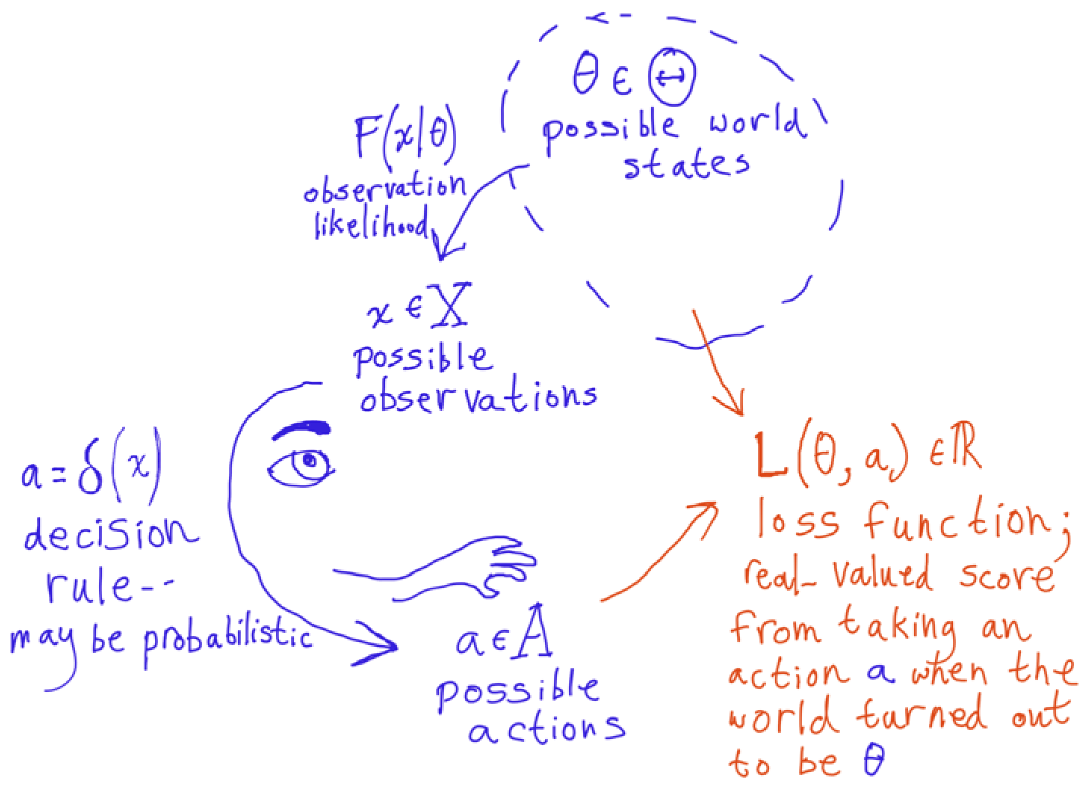
I somewhat jokingly think of Critch's variation as "Critch's Futarchy theorem" -- in the same way that Harsanyi shows that utilitarianism is the unique way to make rational collective decisions when everyone agrees about the facts on the ground, Critch shows that rational collective decisions when there is disagreement must involve a betting market. However, Critch's conclusion is not quite Futarchy. It is more extreme: in Critch's framework, agents bet their voting stake rather than money! The more bets you win, the more control you have over the system; the more bets you lose, the less your preferences will be taken into account. This is, perhaps, rather harsh in comparison to governance systems we would want to implement. However, rational agents of the classical Bayesian variety are happy to make this trade.

Without further adieu, let's dive into the theorems.

Basic CCT

We set up decision problems like this:

- Θ is the set of possible states of the external world.
- X is the set of possible observations.
- A is the set of actions which the agent can take.
- $F(x|\theta)$ is a likelihood function, giving the probability of an observation $x \in X$ under a particular world-state $\theta \in \Theta$.
- D is a set of decision rules. For $\delta \in D$, $\delta(x)$ outputs an action. Stochastic decision rules are allowed, though, in which case we should really think of it as outputting an action probability.
- $L(\theta, a)$, the loss function, takes a world $\theta \in \Theta$ and an action $a \in A$ and returns a real-valued "loss". L encodes preferences: the lower the loss, the better. One way of thinking about this is that the agent knows how its actions play out in each possible world; the agent is only uncertain about consequences because it doesn't know which possible world is the case.



In this post, I'm only going to deal with cases where Θ and X are finite. This is not a minor theoretical convenience -- things get significantly more complicated with unbounded sets, and the justification for Bayesianism in particular is weaker. So, it's potentially quite interesting. However, there's only so much I want to deal with in one post.

Some more definitions:

The ***risk*** of a policy in a particular true world-state: $R(\theta, \delta)=E_{F(x|\theta)}[L(\theta, \delta(x))]$.

A decision rule δ^* is a ***pareto improvement*** over another rule δ if and only if $R(\theta, \delta) \geq R(\theta, \delta^*)$ for all θ, and strictly > for at least one. This is typically called ***dominance*** in treatments of CCT, but it's exactly parallel to the idea of pareto-improvement from economics and game theory: everyone is at least as well off, and at least one person is better off. An improvement which harms no one. The only difference here is that it's with respect to possible states, rather than people.

A decision rule δ is ***admissible*** if and only if there is no pareto improvement over it. The idea is that there should be no reason not to take pareto improvements, since you're only doing better no matter what state the world turns out to be in. (We could also call this *pareto-optimal*.)

A class C of decision rules is a ***complete class*** if and only if for any rule *not* in C, $\delta \notin C$, there exists a rule δ^* in C which is a pareto improvement. Note, not every rule in a complete class will be admissible itself. In particular, the set of all decision rules is a complete class. So, the complete class is a device for proving a weaker result than admissibility. This will actually be a bit silly for the finite case, because we can characterize the set of admissible decision rules. However, it is the namesake of complete class theorems in general; so, I figured that it would be confusing not to include it here.

Given a probability distribution $\pi(\theta)$ on world-states, the ***Bayes risk*** $r(\pi, \delta)$ is the expected risk over worlds, IE: $E_{\pi(\theta)}R(\theta, \delta)$.

A probability distribution π is ***non-dogmatic*** when $\pi(\theta) > 0$ for all θ.

A decision rule is ***bayes-optimal*** with respect to a distribution π if it minimizes Bayes risk with respect to π. (This is usually called a *Bayes rule* with respect to π, but that seems fairly confusing, since it sounds like "Bayes' rule" aka Bayes' theorem.)

THEOREM: *When* Θ *and* A *are finite, decision rules which are bayes-optimal with respect to a non-dogmatic* π *are admissible.*

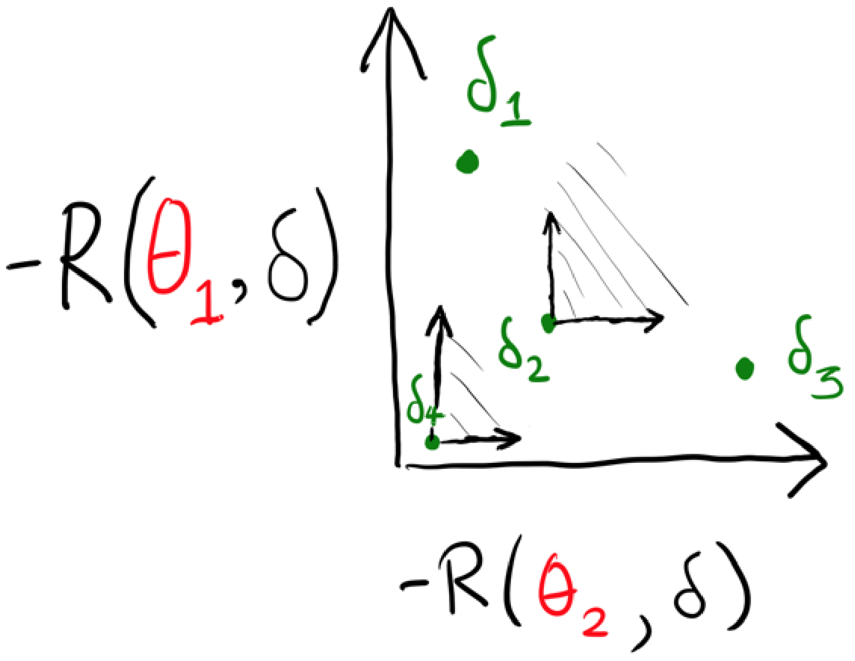
PROOF: On the one hand, if δ is Bayes-optimal with respect to non-dogmatic π, it minimizes the expectation $E_{\pi}(\theta)R(\theta, \delta)$. Since $\pi(\theta) > 0$ for each world, any pareto-improvement $\delta^{'}$ (which must be strictly better in some world, and not worse in any) must decrease this expectation. So, δ must be minimizing the expectation if it is Bayes-optimal. □

THEOREM: (basic CCT) *When* Θ *and* A *are finite, a decision rule* δ *is admissible if and only if it is Bayes-optimal with respect to some prior* π*.*

PROOF: If δ is admissible, we wish to show that it is Bayes-optimal with respect to some π.

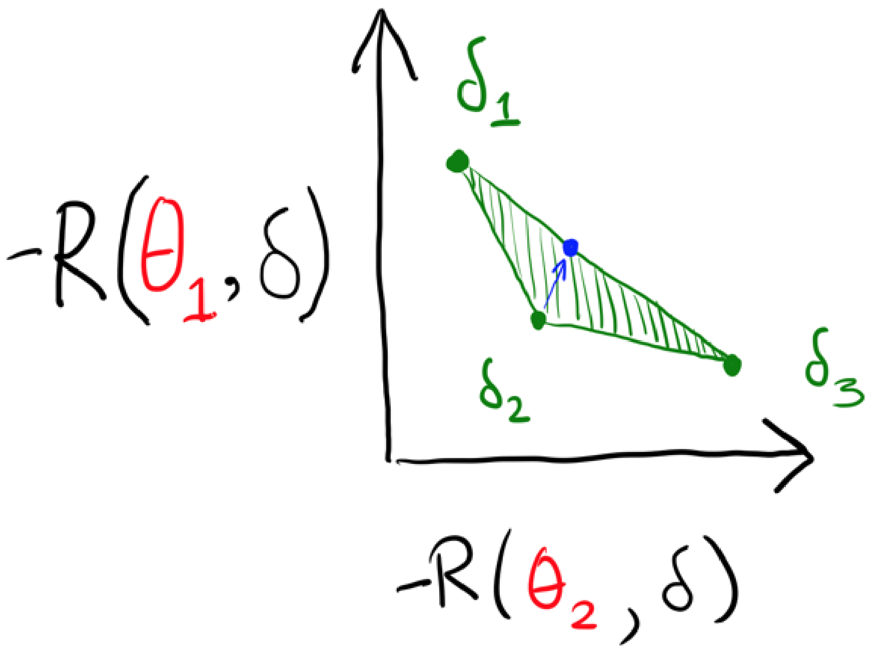
A decision rule has a risk in each world; think of this as a vector in $R^{|\Theta|}$. The set R of achievable risk vectors in $R^{|\Theta|}$ (given by all δ) is convex, since we can make mixed strategies between any two decision rules. It is also closed, since A and X are finite. Consider a risk vector s as a point in this space (not necessarily achievable by any δ). Define the ***lower quadrant*** Q(s) to be the set of points which would be pareto improvements if they were achievable by a decision rule. Note that for an admissible decision rule with risk vector s, Q(s) and R are disjoint. By the hyperplane separation theorem, there is a separating hyperplane H between Q(s) and R. We can define $\pi(\theta)$ by taking a vector normal to the hyperplane and normalizing it to sub to one. This is a prior for which δ is Bayes-optimal, establishing the desired result. □

If this is confusing, I again suggest Peter Hoff's course notes. However, here is a simplified illustration of the idea for two worlds, four pure actions, and no observations:



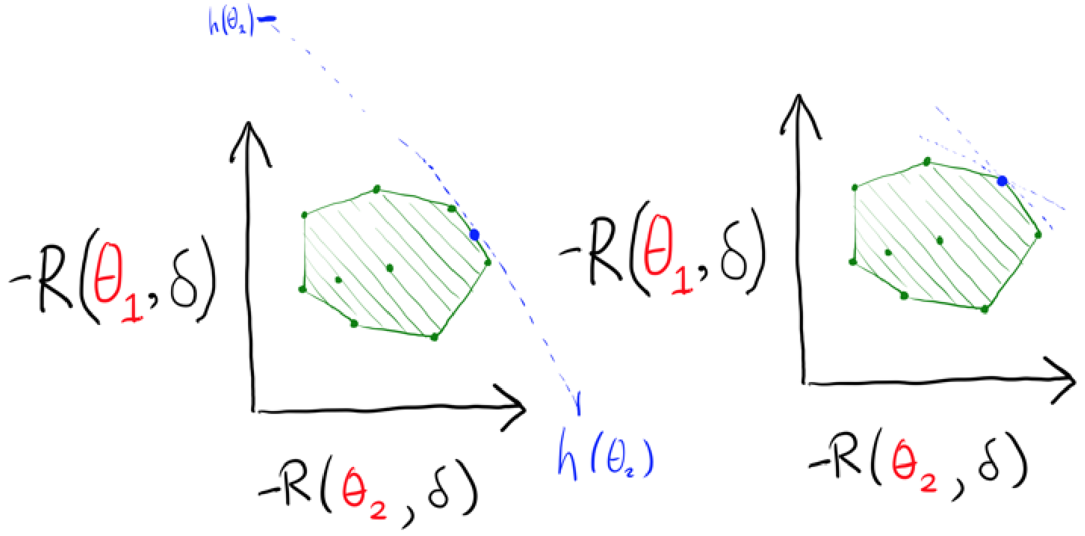
(I used $-R(\theta, \delta)$ because I am more comfortable with thinking of "good" as "up", IE, thinking in terms of utility rather than loss.)

The black "corners" coming from δ_2 and δ_4 show the beginning of the Q() set for those two points. (You can imagine the other two, for δ_1 and δ_3.) Nothing is pareto-dominated except for δ_4, which is dominated by everything. In economics terminology, the first three actions are on the pareto frontier. In particular, δ_2 is not pareto-dominated. Putting some numbers to it, δ_2 could be worth (2,2), that is, worth two in each world. δ_1 could be worth (1,10), and δ_3 could be worth (10,1). There is no prior over the two worlds in which a Bayesian would want to take action δ_2. So, how do we rule it out through our admissibility requirement? We add mixed strategies:



Now, there's a new pareto frontier: the line stretching between δ_1 and δ_3, consisting of strategies which have some probability of taking those two actions. Everything else is pareto-dominated. An agent who starts out considering δ_2 can see that mixing between δ_1 and δ_3 is just a better idea, no matter what world they're in. This is the essence of the CCT argument.

Once we move to the pareto frontier of the set of mixed strategies, we can draw the separating hyperplanes mentioned in the proof:



(There may be a unique line, or several separating lines.) The separating hyperplane allows us to derive a (non-dogmatic) prior which the chosen decision rule is consistent with.

Removing Likelihoods (and other unfortunate assumptions)

Assuming the existence of a likelihood function F is rather strange, if our goal is to argue that agents should use probability and expected utility to make decisions. A purported decision-theoretic foundation should not assume that an agent has any probabilistic beliefs to start out.

Fortunately, this is an extremely easy modification of the argument: restricting F to either be zero or one is just a special case of the existing theorem. This does not limit our expressive power. Previously, a world in which the true temperature is zero degrees would have some probability of emitting the observation "the temperature is one degree", due to observation error. Now, we consider the error a part of the world: there is a world where the true temperature is zero and the measurement is one, as well as one where the true temperature is zero and the measurement is zero, and so on.

Another related concern is the assumption that we have mixed strategies, which are described via probabilities. Unfortunately, this is much more central to the argument, so we have to do a lot more work to re-state things in a way which doesn't assume probabilities directly. Bear with me -- it'll be a few paragraphs before we've done enough work to eliminate the assumption that mixed strategies are described by probabilities.

It will be easier to first get rid of the assumption that we have cardinal-valued loss L. Instead, assume that we have an ordinal preference for each world, $\delta_1 <_\theta \delta_2$. We then apply the VNM theorem within each θ, to get a cardinal-valued utility within each world. The CCT argument can then proceed as usual.

Applying VNM is a little unsatisfying, since we need to assume the VNM axioms about our preferences. Happily, it is easy to weaken the VNM axioms, instead letting the assumptions from the CCT setting do more work. A detailed write-up of the following is being worked on, but to briefly sketch:

First, we can get rid of the independence axiom. A mixed strategy is really a strategy which involves observing coin-flips. We can put the coin-flips inside the world (breaking each θ into more sub-worlds in which coin-flips come out differently). When we do this, the independence axiom is a consequence of admissibility; any violation of independence can be undone by a pareto improvement.

Second, having made coin-flips explicit, we can get rid of the axiom of continuity. We apply the VNM-like theorem from the paper Additive representation of separable preferences over infinite products, by Marcus Pivato. This gives us cardinal-valued utility functions, but without the continuity axiom, our utility may sometimes be represented by infinities. (Specifically, we can consider surreal-numbered utility as the most general case.) You can assume this never happens if it bothers you.

More importantly, at this point we don't need to assume that mixed strategies are represented via pre-existing probabilities anymore. Instead, they're represented by the coins.

I'm fairly happy with this result, and apologize for the brief treatment. However, let's move on for now to the comparison to social choice theory I promised.

Utilitarianism

I said that θ_i are "possible world states" and that there is an "agent" who is "uncertain about which world-state is the case" -- however, notice that I didn't really *use* any of that in the theorem. What matters is that for each θ, there is a preference relation on actions. CCT is actually about compromising between different preference relations.

If we drop the observations, we can interpret the θ_i as *people*, and the A as potential collective actions. The δ are potential social choices, which are admissible when they are pareto-efficient with respect to individual's preferences.

Making the hyperplane argument as before, we get a π which places positive weight on each individual. This is interpreted as each individual's weight in the coalition. The collective decision must be the result of a (positive) linear combination of each individual's cardinal utilities -- and those cardinal utilities can in turn be constructed via an application of VNM to individual ordinal preferences. This result is very similar to Harsanyi's utilitarianism theorem.

This is not only a nice argument for utilitarianism, it is also an amusing mathematical pun, since it puts utilitarian "social utility" and decision-theoretic "expected utility" into the same mathematical framework. Just because both can be derived via pareto-optimality arguments doesn't mean they're necessarily the same thing, though.

Harsanyi's theorem is not the most-cited justification for utilitarianism. One reason for this may be that it is "overly pragmatic": utilitarianism is about *values;* Harsanyi's theorem is about *coherent governance*. Harsanyi's theorem relies on imagining a collective decision which has to compromise between everyone's values, and specifies what it must be like. Utilitarians don't imagine such a global decision can really be made, but rather, are trying to specify their own altruistic values. Nonetheless, a similar argument applies: altruistic values are enough of a "global decision" that, hypothetically, you'd want to run the Harsanyi argument if you had descriptions of everyone's utility functions and if you accepted pareto improvements. So there's an argument to be made that that's still what you want to approximate.

Another reason, mentioned by Jessicata in the comments, is that utilitarians typically value egalitarianism. Harsanyi's theorem only says that you must put *some* weight on each individual, not that you have to be *fair.* I don't think this is much of a problem -- just as CCT argues for "some" prior, but realistic agents have further considerations which make them skew towards maximally spread out priors, CCT in social choice theory can tell us that we need *some* weights, and there can be extra considerations which push us toward egalitarian weights. Harsanyi's theorem is still a strong argument for a big chunk of the utilitarian position.

Futarchy

Now, as promised, Critch's 'futarchy' theorem.

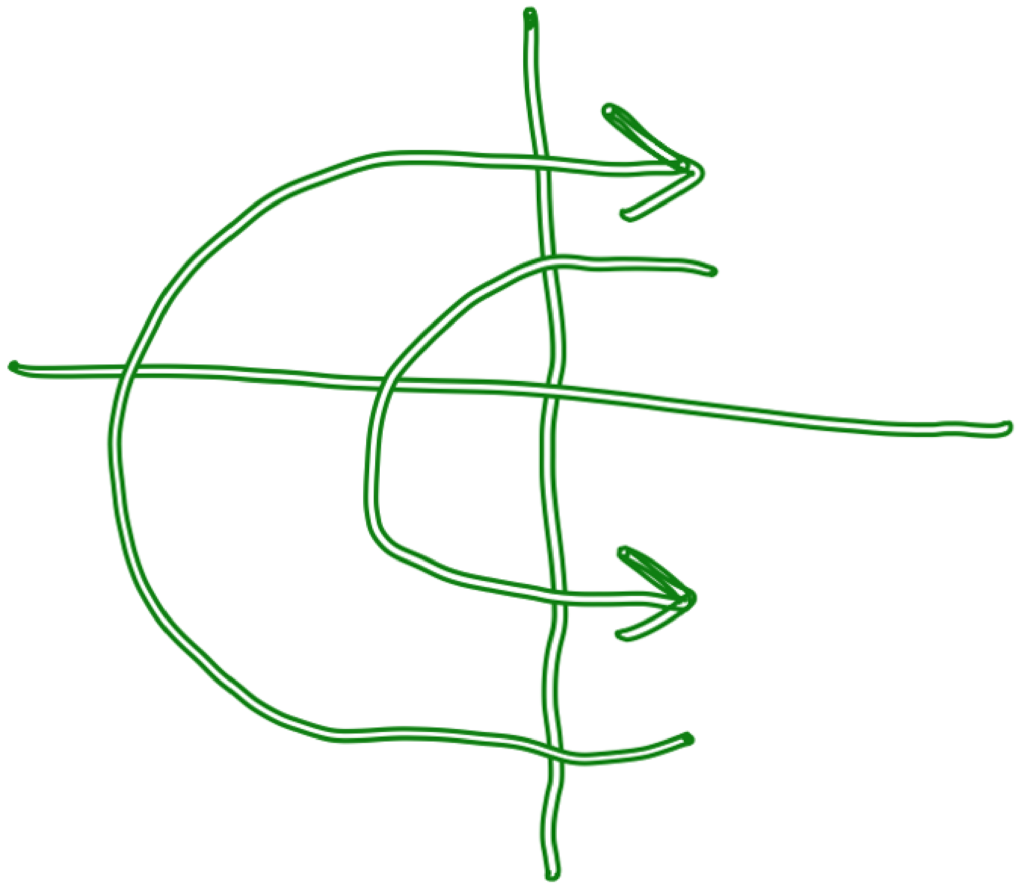
If we add observations back in to the multi-agent interpretation, $F(x|\theta)$ associates each agent with a probability distribution on observations. This can be interpreted as each agent's beliefs. In the paper Toward Negotiable Reinforcement Learning, Critch examined pareto-optimal sequential decision rules in this setting. Not only is there a function π which gives a weight for each agent in the coalition, but *this* π *is updated via Bayes' Rule as observations come in.* The interpretation of this is that the agents in the coalition want to bet on their differing beliefs, so that agents who make more correct bets gain more influence over the decisions of the coalition.

This differs from Robin Hanson's futarchy, whose motto "*vote on values, but bet beliefs*" suggests that everyone gets an equal vote -- you lose *money* when you bet, which loses you influence on *implementation* of public policy, but you still get an equal share of *value.* However, Critch's analysis shows that Robin's version can be strictly improved upon, resulting in Critch's version. (Also, Critch is not proposing his solution as a system of governance, only as a notion of multi-objective learning.) Nonetheless, the spirit still seems similar to Futarchy, in that the control of the system is distributed based on bets.

If Critch's system seems harsh, it is because we wouldn't really want to bet away all our share of the collective value, nor do we want to punish those who would bet away all their value too severely. This suggests that we (a) just *wouldn't* bet everything away, and so wouldn't end up too badly off; and (b) would want to still take care of those who bet their own value away, so that the consequences for those people would not actually be so harsh. Nonetheless, we can also try to take the problem more seriously and think about alternative formulations which seem less strikingly harsh.

Conclusion

One potential research program which may arise from this is: take the analogy between social choice theory and decision theory very seriously. Look closely at more complicated models of social choice theory, including voting theory and perhaps mechanism design. Understand the structure of rational collective choice in detail. Then, try to port the lessons from this back to the individual-agent case, to create decision theories more sophisticated than simple Bayes. Mirroring this on the four-quadrant diagram from early on:



And, if you squint at this diagram, you can see the letters "CCT".

(Closing visual pun by Caspar Österheld.)

Clarifying Consequentialists in the Solomonoff Prior

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

I have spent a long time being confused about Paul’s post on consequentialists in the Solomonoff prior. I now think I understand the problem clearly enough to engage with it properly.

I think the reason I was confused is to a large degree a problem of framing. It seemed to me in the course of discussions I had to deconfuse myself to me that similar confusions are shared by other people. In this post, I will attempt to explain the framing that helped clarify the problem for me.

i. A brief sketch of the Solomonoff prior

The Solomonoff, or Universal, prior is a probability distribution over strings of a certain alphabet (usually over all strings of 1s and 0s). It is defined by taking the set of all Turing machines (TMs) which output strings, assigning to each a weight proportional to

(where L is its description length), and then assigning to each string a probability equal to the weights of the TMs that compute it. The description length is closely related to the amount of information required to specify the machine; I will use description length and amount of information for specification interchangeably.

(The actual formalism is in fact a bit more technically involved. I think this picture is detailed enough, in the sense that my explanation will map onto the real formalism about as well.)

The above defines the Solomonoff prior. To perform Solomonoff induction, one can also define conditional distributions by considering only those TMs that generate strings beginning with a certain prefix. In this post, we’re not interested in that process, but only in the prior.

ii. The Malign Prior Argument

In the post, Paul claims that the prior is dominated by consequentialists. I don’t think it is quite dominated by them, but I think the effect in question is plausibly real.

I’ll call the key claim involved the Malign Prior Argument. On my preferred framing, it goes something like this:

Premiss: For some strings, it is easier to specify a Turing Machine that simulates a reasoner which decides to predict that string, than it is to specify the intended generator for that string.

Conclusion: Therefore, those strings' Solomonoff prior probability will be dominated by the weight assigned to the TM containing the reasoner.

It's best to explain the idea of an 'intended generator' with examples. In the case of a camera signal as the string, the intended generator is something like a TM that simulates the universe, plus a specification of the point in the simulation where the camera input should be sampled. Approximations to this, like a low-fidelity simulation, can also be considered intended generators.

There isn't anything special about the intended generator's relationship to the string - it's just one way in which that string can be generated. It seems most natural to us as humans, and the Occamian nature of SI feels like it should be biased towards such strings, but nothing in principle stops something less 'natural' from being in fact a shorter description.

This idea of 'naturalness' is important in understanding what the Malign Prior Argument is about; I will use it roughly to refer to something like 'the set of Turing Machines that don't involve reasoners that attempt to influence the prior', or 'the set of intended generators'. It's vague, but I think it gets across the point.

I read most of Paul's post as an existence argument for the premiss, using consequentialists in other worlds as the reasoners. I don't think all such reasoners are like Paul describes; I also doubt that all or even most strings are subject to this effect, but find it very plausible that some are.

I think the argument is not, at its core, about these reasoners making the strings they output *more likely* than the 'true string'. It is concerning enough that there is any effect *at all* that these reasoners have on the prior, which is the fact this argument establishes.

As a side note, it's also worth noting that this is not about these reasoners breaking out of the box and taking over our world, although that is also a related concern one might have.

iii. The support for premiss 1

Consider a string S' with very high natural K-complexity (description length of the intended generator) that shares a prefix with a string S that is of high interest to human-like civilisations.

I claim that the prior probability of this string is higher than it 'naturally' 'should' be, in the sense that a large part of the weight that composes this probability is coming from a TM that simulates a reasoner that is attempting to influence the prior.

The reasons this happens are:

1. A reasoner in a TM can have an arbitrarily long amount of compute time to decide what strings to output.
2. Specifying reasoners is cheap relative to specifying the string S'.
3. There exists a reasoner whose goals are best served by influencing the prior to make S' more likely.

1 is a crucial property of the Solomonoff prior that allows this to happen. A TM in the Solomonoff prior can think for a very, very long time — enough to e.g. simulate an

Ackerman(Ackerman(10)) initial world states each for Ackerman(Ackerman(10)) timesteps. It can perform something close to an exhaustive search of all possible civilizations and decide to attempt to influence the one that is most susceptible to be influenced, if that's what it wants to do. This is a ridiculous computation, but we're talking about a mathematical object, not an actual process that we run. It's plausible that if the prior was also weighted by speed of computation, these effects would be far less pronounced (and maybe would not arise at all).

To see that 2 and 3 are plausible, we need to think about S', which by assumption is a string with high natural K-complexity. This high complexity 'buys' us the space to specify a reasoner, and the space to specify values, without making the TM more complex than a natural generator of S'. Now, because S is by assumption of interest to civilisations, there likely exists a TM containing a reasoner that performs its exhaustive search, finds S, and concludes that its values are best served by making S' more likely (e.g. to influence the decision-making of civilisations that are thinking about what S is, given a prefix of it known to them).

In a way, this agent uses its simplicity to give more simplicity to some other string. That is how the prior gets hijacked.

Note that this reasoner will need to have goals that are simpler than the natural generator of S' in order to actually contribute to S' being more likely - otherwise, specifying its TM would be more expensive than specifying the natural generator of S'.

The above is non-constructive (in the mathematical sense), but nevertheless the existence of strings S' that are affected thus seems plausible. The spaces of possible TMs and of the strings we (or other users of the Solomonoff prior) could be interested in are simply too vast for there not to be such TMs. Whether there are very many of these, or whether they are so much more complicated than the string S so as to make this effect irrelevant to our interests, are different questions.

iv. Alien consequentialists

In my view, Paul's approach in his post is a more constructive strategy for establishing 2 and 3 in the argument above. If correct, it suggests a stronger result - not only does it cause the probability of S' to be dominated by the TM containing the reasoner, it makes the probability of S' roughly comparable to S, for a wide class of choices of S.

In particular, the choice of S that is susceptible to this is something like the camera example I used, where the natural generator is S is a specification of our world together with a location where we take samples from. The alien civilisation is a way to construct a Turing Machine that outputs S' which has comparable complexity to S.

To do that, we specify a universe, then run it for however long we want, until we get somewhere within it smart agents that decide to influence the prior. Since 1 is true, these agents have an arbitrary amount of time to decide what they output. If S is important, there probably will be a civilisation somewhere in some simulated world which will decide to attempt to influence decisions based on S, and output an appropriate S'. We then specify the output channel to be whatever they decide to use as the output channel.

This requires a relatively modest amount of information - enough to specify the universe, and the location of the output. This is on the same order as the natural generator for S itself, if it is like a camera signal.

Trying to specify our reasoner within this space (reasoners that naturally develop in simulations) does place restrictions on what kind of reasoner we up end up with. For instance, there are now some implicit runtime bounds on many of our reasoners, because they likely care about things other than the prior. Nevertheless, the space of our reasoners remains vast, including unaligned superintelligences and other odd minds.

v. Conclusion. Do these arguments actually work?

I am mostly convinced that there is at least some weirdness in the Solomonoff prior.

A part of me wants to add ‘especially around strings whose prefixes are used to make pivotal decisions’; I’m not sure that is right, because I think scarcely anyone would actually use this prior in its true form - except, perhaps, an AI reasoning about it abstractly and naïvely enough not to be concerned about this effect despite having to explicitly consider it.

In fact, a lot of my doubt about the malign Solomonoff prior is concentrated around this concern: if the reasoners don’t believe that anyone will act based on the true prior, it seems unclear why they should spend a lot of resources on messing with it. I suppose the space is large enough for at least some to get confused into doing something like this by mistake.

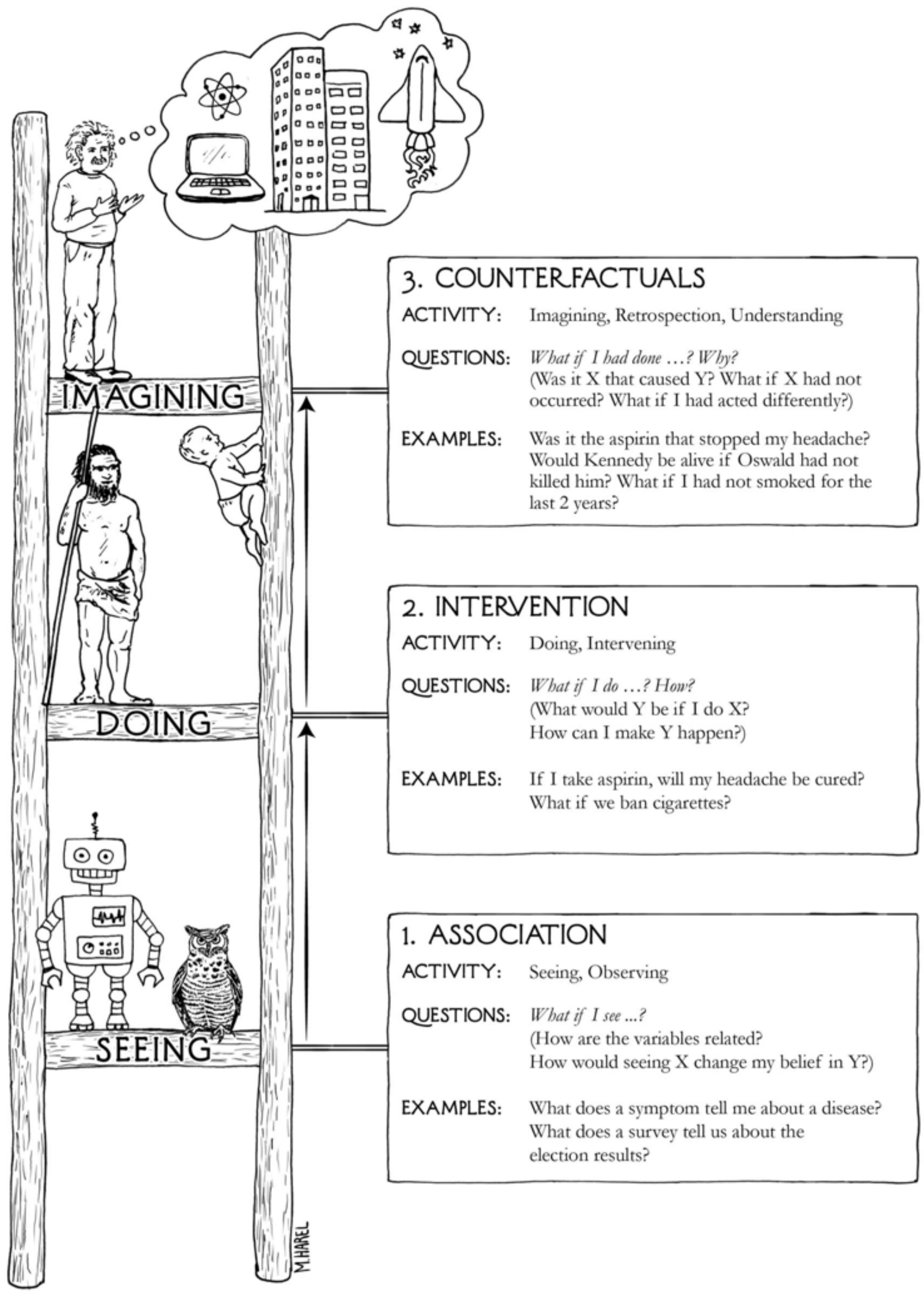
I think that even if my doubts are correct, there will still be weirdness associated with the agents that are specified directly, along the lines of section iii, if not those that appear in simulated universes, as described in iv.

On the Role of Counterfactuals in Learning

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

The following is a hypothesis regarding the purpose of counterfactual reasoning (particularly in humans). It builds on Judea Pearl's three-rung Ladder of Causation (see below).

One important takeaway from this hypothesis is that counterfactuals really only make sense in the context of computationally bounded agents.


3. COUNTERFACTUALS
ACTIVITY: Imagining, Retrospection, Understanding
QUESTIONS: What if I had done …? Why? (Was it X that caused Y? What if X had not occurred? What if I had acted differently?)
EXAMPLES: Was it the aspirin that stopped my headache? Would Kennedy be alive if Oswald had not killed him? What if I had not smoked for the last 2 years?
2. INTERVENTION
ACTIVITY: Doing, Intervening
QUESTIONS: What if I do …? How? (What would Y be if I do X? How can I make Y happen?)
EXAMPLES: If I take aspirin, will my headache be cured? What if we ban cigarettes?
1. ASSOCIATION
ACTIVITY: Seeing, Observing
QUESTIONS: What if I see …? (How are the variables related? How would seeing X change my belief in Y?)
EXAMPLES: What does a symptom tell me about a disease? What does a survey tell us about the election results?
IMAGINING
DOING
SEEING
M. HAREL

(Figure taken from *The Book of Why* [Chapter 1, p. 6].)

Summary

Counterfactuals provide initializations for use in MCMC sampling.

Preliminary Definitions

Association (model-free):

$$\Pr(Y = y \mid X = x)$$

Intervention/Hypothetical (model-based):

$$\Pr(Y = y \mid do(X = x))$$

Counterfactual (model-based):

$$\Pr(Y = y \mid do(X = x), Y = y')$$

In the counterfactual, we have already observed an outcome y' but wish to reason about the probability of observing another outcome y (possibly the same as y') under do(X = x).

Note: Below, I use the terms "model" and "causal network" interchangeably. Also, an "experience" is an observation of a causal network in action.

Assumptions

1. Real-world systems are highly complex, often with many causal factors influencing system dynamics.
2. Humans minds are computationally bounded (in time, memory, and precision).
3. Humans do not naturally think in terms of continuous probabilities; they think in terms of discrete outcomes and their relative likelihoods.

Relevant Literature:

Lieder, F., Griffiths, T. L., Huys, Q. J., & Goodman, N. D. (2018). The anchoring bias reflects rational use of cognitive resources. *Psychonomic bulletin & review*, *25*(1), 322-349.

Sanborn, A. N., & Chater, N. (2016). Bayesian brains without probabilities. *Trends in cognitive sciences*, *20*(12), 883-893.

Theory

Claim 1.

From a notational perspective, in going from a hypothetical to a counterfactual, the generalization lies solely in the ability to reason about a concrete scenario *starting from an alternative scenario* (the counterfactual). In theory, given infinite computational resources, the do-operator can, on its own, reason forward about anything by considering only hypotheticals. Thus, a counterfactual would be an inadmissible object under such circumstances. (Perfect knowledge of the system is not required if one can specify a prior. All that is required is sufficient computational resources.)

Corollary 1.1.

Counterfactuals are only useful when operating with limited computational resources, where "limited" is defined relative to the agent doing the reasoning and the constraints they face (e.g., limited time to make a decision, inability to hold enough items in memory, and any such combinations of these constraints).

Corollary 1.2.

If model-based hypothetical reasoning (i.e. "simulating") is a sufficient tool to resolve all human decisions, then all of our experiences/observations should go toward building a model that is as accessible and accurate as possible, given our computational limitations.

By **Assumption 1**, the vast majority of human decision-making theoretically consists in reasoning about a "large" number of causal interactions at once, where “large” here means an amount that is beyond the bounds of the human mind (**Assumption 2**). Thus, by **Claim 1**, we are in the regime where counterfactuals are useful. But in what way are they useful?

By **Corollary 1.2**, we wish to build a useful model based upon our experiences. A useful model is one that is as predictively accurate as possible while still being accessible (i.e. interpretable) by the human mind. Given that: (1) a model is describable as data, (2) the most data can be stored in our brains in the form of long-term memory, and (3) the maximal predictive accuracy of a model is a non-decreasing function of its description length, then a maximally predictive model is one that is stored in our long-term memory. However, human working memory is limited in capacity relative to long-term memory.

Claim 2.

The above are competing factors: A more descriptive (and predictive) model (represented by more data) may fit in long-term memory, but due to a limited working memory, it may be inaccessible (at least in a way that leverages its full capabilities). Thus, attentional mechanisms are required to guide our retrieval of subcomponents of the full model to load into working memory.

Again, by **Assumptions 1, 2**, our models are approximate — both inaccurate and incomplete. Thus, we wish to improve our models by integrating over our entire experiences. This equates to computing the following posterior distribution:

Pr(causal network | experience)

$$= \frac{\Pr(\text{experience} \mid \text{causal network})}{\Pr(\text{experience})} \times \Pr(\text{causal netw}$$

By **Assumption 3**, humans cannot compute updates to their priors according to the above formula.

Claim 3.

Humans do something akin to MCMC sampling to approximate the above posterior. Because MCMC methods (e.g., Gibbs sampling, Metropolis-Hastings) systematically explore the space of models in a *local* and *incremental* manner (e.g., by conditioning on all but one variable in Gibbs sampling, or by taking local steps in model space in Metropolis-Hastings) AND only require reasoning via likelihood ratios (**Assumption 3**), we can overcome the constraints imposed by our limited working memory and still manage to update models that fit in long-term memory but not entirely in working memory.

MCMC methods require initialization (i.e. a sample to start from).

Claim 4.

Counterfactuals provide this initialization. Given that our model is built up entirely of true samples of the world, our aims is to interpolate between these samples. (We don't really have a prior at birth on the ground-truth causal network on which the world operates.) Thus, we can only trust our model with 100% credibility at observed samples. Furthermore, by **Assumption 2**, we are pressured to minimize time to convergence of any MCMC method. Hence, the best we can do is to begin the MCMC sampling procedure starting from a point that we *know* belongs in the support of the distribution (and likely in a region of high density).

From the Metropolis-Hastings Wikipedia:

> Although the Markov chain eventually converges to the desired distribution, the initial samples may follow a very different distribution, especially if the starting point is in a region of low

density. As a result, a *burn-in* period is typically necessary.

Counterfactuals allow us to avoid the need for any costly burn-in phase.

Agents That Learn From Human Behavior Can't Learn Human Values That Humans Haven't Learned Yet

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

[Epistemic status: ¯_(ツ)_/¯]

Armstrong and Mindermann write about a no free lunch theorem for inverse reinforcement learning (IRL): the same action can reflect many different combinations of values and (irrational) planning algorithms.

I think even assuming humans were fully rational expected utility maximizers, there would be an important underdetermination problem with IRL and with all other approaches that infer human preferences from their actual behavior. This is probably obvious if and only if it's correct, and I don't know if any non-straw people disagree, but I'll expand on it anyway.

Consider two rational expected utility maximizing humans, Alice and Bob.

Alice is, herself, a value learner. She wants to maximize her true utility function, but she doesn't know what it is, so in practice she uses a probability distribution over several possible utility functions to decide how to act.

If Alice received further information (from a moral philosopher, maybe), she'd start maximizing a specific one of those utility functions instead. But we'll assume that her information stays the same while her utility function is being inferred, and she's not doing anything to get more; perhaps she's not in a position to.

Bob, on the other hand, isn't a value learner. He knows what his utility function is: it's a weighted sum of the same several utility functions. The relative weights in this mix happen to be identical to Alice's relative probabilities.

Alice and Bob will act the same. They'll maximize the same linear combination of utility functions, for different reasons. But if you could find out more than Alice knows about her true utility function, then you'd act differently if you wanted to truly help Alice than if you wanted to truly help Bob.

So in some cases, it's not enough to look at how humans behave. Humans are Alice on some points and Bob on some points. Figuring out details will require explicitly addressing human moral uncertainty.

Decision-theoretic problems and Theories; An (Incomplete) comparative list

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.
This is a linkpost for https://docs.google.com/spreadsheets/d/1zeVyhwjq97ILSnz7oO7qaoDpV6QCv0k5kmCTDjh3McU/edit?usp=sharing

So, this is a work-in-progress, but the idea is to a) get an eventually exhaustive list of decision-theoretic problems, and b) detail the '''answers''' given by each major decision theory.

(Stretch goals would including listing other kinds of properties, and representing disagreement about the ''answers'')

Proposed amendments and additions are welcomed!

Mathematical Mindset

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

I agree that optimization amplifies things. I also agree that a mathematical mindset is important for AI alignment. I don't, however, think that a "mathematical mindset" is the same as a "proof mindset". Rather, I think that the latter is closer to being a "programming mindset" -- or, indeed, a "security mindset". And that a "mathematical mindset" is largely missing from AI-alignment discourse at present.

Whereas others see a division between two clusters, of the form,

science/physics

vs.

mathematics/programming/logic

I, by contrast, see a hierarchical progression that looks something like:

science < programming < physics < mathematics < logic <...

where, in this context, these words have meanings along the following lines:

science: things being made of parts; decomposition

programming: things being made of moving parts; constant-velocity motion; causal networks

physics: things being made of moving spatial parts; accelerated motion, rotation, fluidity; substance

mathematics: models being made of parts; transubstantiation; metaphysics; theorization

logic: concepts being made of parts; time reversal; ontology

Of course I'm not using these words standardly here. One reason for this is that, in this discussion, no one is: we're talking about *mindsets*, not about sociological disciplines or even clusters of particular ideas or "results".

But the really important reason I'm not following standard usage is because I'm not trying to invoke standard concepts; instead, I'm trying to invent "the right" concepts. Consequently, I can't just use standard language, because standard language implies a model of the world different from the one that I want to use.

It is commonly believed that if you want to introduce a new concept that is similar or related (but--of course--nonidentical) to an old concept, you shouldn't use the same word for the new concept and the old, because that would be "confusing". I wish to explicitly disagree with this belief.

This view presupposes that *actively shifting between models of the world* is not in our repertory of mental operations. But I specifically want it to be!

In fact, I claim that *this*, and not proof, is what a "mathematical mindset" is really about.

For mathematics is not about proofs; it is about *definitions*. The essence of great mathematics is coming up with a powerful definition that results in short proofs.

What makes a definition "powerful" is that it reflects a *conceptual upgrade* -- as distinct from mere conceptual analysis. We're not just trying to figure out what we mean; we're trying to figure out what we *should* mean.

A mathematical definition is what the answer to a philosophical problem looks like. An example I particularly like is the definition of a topological space. I don't know for a fact that this is what people "really meant" when they pondered the nature of "space" during all the centuries before Felix Hausdorff came up with this definition; it doesn't matter, because the power of this definition shows that it is what they should have meant.

(And for that reason, I'm comfortable saying that it *is* what they "meant" -- acknowledging that this is a linguistic fiction, but using it anyway.)

Notably, mathematical definitions are often redefinitions: they take a term already in use and define it in a new way. And, notably, the new definition often bears scant resemblance to the old, let alone any “intuitive” meaning of the term -- despite presenting itself as a successor. This is not a bug. It is what philosophical progress -- theoretical progress, progress in understanding -- looks like. The relationship between the new and the old definitions is explained not in the definitions themselves, but in the relationship between the theories of which the definitions are part.

Having a “mathematical mindset” means being comfortable with words being redefined. This is because it means being comfortable with models being upgraded -- in particular, with models being related and compared to each other: the activity of theorization.

It occurs to me, now that I think about it, that the term “theorization” is not very often used in the AI-alignment and rationalist communities, compared with what one might expect given the degree of interest in epistemology. My suspicion is that this reflects an insufficiency of comfort with the idea of *models* (as opposed to *things*) being made of *parts* (in particular, being made of parameters), such that they are relatable to and transformable into each other.

This idea is, approximately, what I am calling “mathematical mindset”. It stands in contrast to what others are calling “mathematical mindset”, which has to do with proofs. The relationship, however, is this: both of them reflect an interest in understanding what is going on, as opposed to merely being able to describe what is going on.

Monk Treehouse: some problems defining simulation

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

When does one program simulate another? When does one program approximately simulate another?

In some contexts, these questions have concrete answers. In other contexts they feel about the same as the problem of functionalist triviality. When simulation comes up in contexts relevant to agent foundations, it tends to feel like the latter. Here, I outline some arguments which I've used in the past to show that a given concept of simulation doesn't quite accomplish what it sets out to do.

For sake of concreteness, I will give two problems which we can imagine solving with an appropriate definition of simulation.

First, consider the problem of taking counterfactuals in order to evaluate consequences of our actions. We may wish to seek copies of our own algorithm in the environment in order to counterfact on them (or to make a mere assumption that their output will be the same as ours). Here, I assume we're trying to solve this problem by looking for sub-components (or properties) of some world model which meet some criterion; namely, forming a simulation of us, according to some formal definition. We've already decided on a program P which represents us (possibly more than one, but we take interest in one at at time), and we are comparing P to (our models of) the environment in some sense. We expect to find one copy of P — ourselves — and we are looking for others.

As a second example, suppose that we have in hand a mathematical description of an AI which we are considering building. Suppose that the AI has various safety and transparency properties, but all these properties are stated in terms of the agent's beliefs and actions, where beliefs are stored in some specified format and actions are output via a specified channel. We are interested in knowing whether building the agent will lead to any "emergent subagents", and if so, whether those agents will be aligned with the original agent (and with us), and what safety and transparency properties those agents might have. In pursuit of this goal, we are attempting to prove theorems about whether certain programs P representing subagents can be simulated on certain sub-components (or properties) of our AI.

A Strawman

Again for sake of concreteness, I'll put forward a straw definition of simulation. Suppose the program P we're searching for simulations of is represented as a Turing machine, and we enrich it with an intuitive set of counterfactuals (moving the head, changing the state, changing symbols anywhere on the tape; all of this at any time step). We thus get a causal graph (the program graph), in which we think of some vertices as input and some as output. I assume our model of the world (in the first example) or the AI (in the second) is similarly made up of components which have states and admit fairly straightforward local counterfactuals, such as removing certain groups of atoms or changing the contents of certain registers. We can thus think of the environment or the AI as a causal graph (the target graph). We claim some part of the target graph simulates the program graph if there is a mapping between the two which successfully represents all the counterfactuals. Since the target graph may be highly redundant, we allow single vertices in the program graph to correspond to collections of vertices in the target graph, and we similarly allow single states in the program graph to correspond to multiple possible states in the target graph. (Imagine a collection of voltages which all map to "0".)

This definition likely has many problems. For example, in a Turing machine virtually all causation has to move through the read/write head, and we have counterfactuals which can move the head. We probably don't care if the environment implements the computation in a way which runs all causation through one point like this. This definition of simulation may also be trivial (in the Functionalist Triviality sense) when the target graph is large enough and rich enough; a target graph with enough counterfactuals will contain some of the right shape to match the program graph. For the sake of this post, just imagine someone has taken this definition as a starting point and attempted to remove some obvious problems. (The arguments I'm presenting were originally replies to a couple different proposals of this general type.)

Monk Treehouse

An order of monks has dedicated themselves to calculating the program P. The monks live in a gigantic treehouse, and have constructed a large, abacus-like computer on which to run P. Large ceremonial stone weights are moved from one place to another in order to execute the algorithm. The weights are so heavy that moving any one of them could cause the treehouse to become unbalanced and fall down. In order to do their computation, the monks have come up with a balancing system. As a first example, I'll suppose that these monks came up with a second program P' when they first built the treehouse, and P' is executed on a separate set of stone weights on the other side of the treehouse. The program P' is different from P, but the monks were able to prove a theorem guaranteeing that the distributions of weights would balance.

Intuitively, the monks on one side of the treehouse are simulating the program P — it's actually what they set out to do. But none of the counterfactuals which we want to take on the program graph map easily to the treehouse weights. If we move a weight on one side to correspond to editing a symbol on the Turing machine's tape, then the treehouse falls over. If we move some weight on the other side to keep things balanced, the counterfactual may temporarily succeed, but the program P' will stop obeying the monks' foreordained guarantee; essentially P' may get thrown off course and fail to counterbalance the next move correctly, destroying the treehouse. If we just add hidden weights to counterbalance our change, the hidden weights could become wrong later.

When such a program P' exists, the treehouse argument is basically pointing out that the "straightforward local counterfactuals" we put in the target graph were not enough; there can be "logical entanglements" which keep us from successfully pointing at the embedding of P.

Of course, we could include in our mapping between program graph and target graph that corresponding to any counterfactual part of the program graph, we create a physical structure which props up the tree within the target graph. This is beginning to become suspicious since the states used for counterfactuals look so much different than the states corresponding to a non-counterfacted run of the program. One thing we want to avoid is having a notion of simulation which could consider an empty patch of air to be computing P, simply by mapping counterfactuals to a case where a computer suddenly appears. (Such a problem would be even worse than the usual functionalist triviality problem.) When we say that a state of some part of the Turing machine corresponds to some state in the target graph, we should mean the same thing whether we're in a counterfactual part of the graphs or not.

The "computer from thin air" problem actually seems very similar to the Monk Treehouse problem. One way of trying to intuitively argue that the monks are executing the program P would be to suppose that all the monks switch to using lightweight tokens to do their computation instead of the heavy stones. This seems to be almost as thorough a change as making a counterfactual computer in empty air. Yet it seems to me that the monk treehouse is really computing P, whereas the air is not.

Suppose we have a definition of simulation which can handle this; that is, somehow the treehouse doesn't fall over under counterfactuals but the notion of simulation doesn't build computers of thin air. It seems to me there are still further complications along the same lines. These monks are good mathematicians, and have devoted their lives to the faithful execution of the program P. One can only imagine that they will be keeping careful records and double checking that every stone is moved correctly. They also will look for any mathematical properties of the process which can be proven and used to double check. When we attempt to map counterfactual Turing machine states to counterfactual stone positions, we will be pinned down by a complicated verification process;. If we just move the stone, the monks will notice it and move it back; if we try to change the monks' beliefs too, then they will notice contradictions in their mathematics, lose faith, and abandon the calculations.

What might work would be to find every single case where a monk calculates the stone position, both before and after the actual stone movement, and map counterfactuals on the program graph to a counterfactual where every single one of these calculations is changed. This likely has problems of its own, but it's interesting as a suggestion that in order to find one simulation of the program P, you need to track down basically all simulations of P (or at least all the ones that might interact).

To make my analogy clear: both emergent subagents within an AI and instances of one's decision algorithm in the environment could be embedded in their surroundings in ways not easily teased out by "straightforward local counterfactuals". Nonetheless, the notion of simulation here seems intuitive, and I can't help but imagine there's an appropriate definition somewhere just beyond ones I've seen or tried. I'd be happy to look at any suggestions.

An Agent is a Worldline in Tegmark V

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

If asked to define what an *agent* is, my usual answer -- one of them, anyway -- is “a worldline in Tegmark V”.

The Tegmark Level V Multiverse (the “V” here is a Roman numeral) is not defined by Max Tegmark (whose hierarchy goes only up to IV), but, as used in agent-foundations circles, it refers to the collection of not-necessarily-consistent mathematical universes, a.k.a. “impossible possible worlds”. It thus contains worlds in which 1+1 = 3, worlds in which triangles have four sides, and perhaps worlds with married bachelors -- in addition, of course, to all the more “ordinary” worlds in Tegmark Levels I-IV (and thus, in particular, us).

This definition of “agent” is intended to evoke the concept of an *observer* in physics (especially relativity), which is a worldline in physical spacetime. “Observer” is a more passive word than “agent”, corresponding to the fact that in physics, worldlines are determined by equations (the “equations of motion”) that represent the laws of physics; whereas the idea in *agent theory* is that the corresponding equations -- those that determine worldlines -- represent the preferences, or “caring structure”, of some entity other than (at any rate, not necessarily identical to) the physical universe.

(I am deliberately avoiding the terms “values” and “goals”, for obscure theoretical reasons that I won’t explain here.)

A worldline in Tegmark IV (to say nothing of V) would, almost by its very nature, suggest a higher degree of “agency” than the ordinary sort of worldline, because it would allow for the possibility that the “observer” or “agent” moves between universes with differing laws of physics. Were we ever to acquire the capacity of “hacking into” our universe and changing its physical laws, for example, this would be the sort of mathematical setting in which our activities would be appropriately modeled. The equations governing our trajectories in that case would be reasonably termed “laws of metaphysics” -- or, indeed (in the most general case at least), “laws of mathematics”.

Importantly, note that this setting enables us to *reason counterfactually about physical laws* (that is, about “metaphysical location”), in exactly the same way that ordinary physics allows us to reason counterfactually about physical location (e.g., “if I were *here* at *this* time, then I would be *there* at *that* time”).

The next step in this progression, Tegmark-V worldlines, might seem absurd at first: hacking into the laws of mathematics, and traveling into the world where 1+1=3? But the subject of *logical uncertainty*, and in particular *logical counterfactuals*, has indeed been receiving attention lately. Essentially, it arises out of situations in which agents need to construct *models* -- that is, maps that aren’t the territory -- not only of the physical world, but of mathematical truths: for example, to reason about how their own or other agents’ “source code” works. In other words, situations in which agents reason about agency.

Doing so is tantamount to reasoning counterfactually about the laws of mathematics. Consequently, reasoning about agency -- something that we would like agents to be able to do, as a result of our having done so -- is equivalent to reasoning about the "physics" of Tegmark V: meta-metaphysics, or metamathematics.

Generalized Kelly betting

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

[tl;dr: It's a mess, don't go there]

[Thanks to Diff for proof reading this post]

Kelly betting has the very nice property that if a gambler is betting according to a given world model, and the amount of money the gambler starts out with equals to the prior probability of that model, then after each round of bets, this gamblers money will equal the current posterior probability.

The problem with Kelly betting is that it relies on only being given one bet at a time, and that the previous bet will be evaluated before you are asked to bet on a new question. Compare this to the situation faced by the traders in a Logical Inductor, where there are always multiple bets every round and the traders don't know when any bet will be settled.

I have (almost) calculated the generalized Kelly criterion, in the case of two dual outcome simultaneous bets, with general market odds and general gambler beliefs. The only remaining part of this calculation is a cubic equation.

Send me an e-mail (linda.linsefors@gmail.com) if you want my notes. There is no guarantee that I will read all blog post comments.

Solving this last equation for general market odds, is in principle not very hard. You can find the general solution to cubic equations on Wikipedia. Except in this specific case the equation is:

$$a(\beta_{11})^3 + b(\beta_{11})^2 + c\beta_{11} + d = 0$$

$$a = m_1 m'_1 (1 - m_1 - m'_1)$$

$$b = m_1 m'_1 - p_1 m_2 m'_1 - p'_1 m_1 m'_2 + 2 p_1 p'_1 m_1 m'_1$$

$$c = p_1 p'_1 [- (m_1 + m'_1) + p_1 m'_2 + p'_1 m_2 - p_1 p'_1]$$

$$d = (p_1 p'_1)^2$$

solve for β_{11}.

The probability that I will get this right by hand is near zero. Maybe someone with Mathematica, or a similar tool could help me?

For the notation, there are two simultaneous and independent bets. Each bet has two outcomes, denoted by index $x \in \{1, 2\}$.

p_x = probability of outcome x, according to gambler.

m_x = probability of outcome x, according to the market.

And the ' superscript denotes the second bet. By definition

$$p_1 + p_2 = 1, \qquad p'_1 + p'_2 = 1$$

$$m_1 + m_2 = 1, \qquad m'_1 + m'_2 = 1$$

β_{11} is just a help variable I made up. It does not have a super clear interpretation, but just happens to be the key to calculating everything else.

As mentioned above, I don't have the final formula for the generalized Kelly criterion, in the case of two dual outcome simultaneous bets, with completely general market odds and gambler beliefs, not until someone solves the above equation. What I currently do have is the special case where the market probabilities for both statements is 50%, and this special case already shows that the nice property of money representing probability, **is not preserved**.

The Bayesian updating factor for hypotheses H, given two independent observations O and O', and $P(O) = P(O') = \frac{1}{2}$, should be

$$\frac{P(O|H)}{P(O)}\frac{P(O'|H)}{P(O')} = 4P(O|H)P(O'|H)$$

The update factor for the amount of money that a gambler following hypothesis H has, given the above circumstances, is

$$\frac{2P(O|H)P(O'|H)}{P(O|H)P(O'|H) + P(\neg O|H)P(\neg O'|H)}$$

Conceptual problems with utility functions, second attempt at explaining

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

[Followup to: Conceptual problems with utility functions]

The previous post was me flailing my arms about wildly trying to gesture at something. After some conversations I think I have a somewhat better idea of what it is and can try to explain it somewhat more systematically. Let us see if it works.

The point that I want to make clear I agree with, is that there is a difference between "valuing fairness" (for example) because you want everyone to get something, versus "valuing fairness" for tactical reasons. For example, suppose that you are playing the Ultimatum Game where the point is to divide the profit from a joint project, and each person thinks they put in more effort than the other and so deserves a larger share. Then maybe you think that the "fair outcome" is for you to get 60%, while the other person thinks that the "fair outcome" is for you to get 40%.

I think a reasonable way to deal with this scenario is: each of you plays a mixed strategy with the end result that on average, both players will get 40% (meaning that 20% of the time the players will fail to reach agreement and nobody will get anything). The nice things about this strategy are (A) it gives both players at least some money and (B) it satisfies a "meta-fairness" criterion that agents with more biased notions of fairness aren't able to exploit the system to get more money out of it.

So now in our decisionmaking process there are two notions of "fairness": the initial notion of fairness that made us decide that the fair split is 60/40, and the "meta-fairness" or "inexploitability" that made us sacrifice some of the other person's money so that we prefer 40/40 to 40/60. Certainly these are different. But my point is that they are not as different as they appear, and I am not sure that there is a principled way of separating your decisionmaking process into two components "utility function" and "decision theory". Moreover, even if there is a clean way of making such a separation, it is not clear to me that "meta-fairness" is a much simpler concept than "object-level fairness". And it seems to me that the utility functions/game theory framework presumes not only that it is a simpler concept, but that it is somehow conceptually equivalent to the concept of "maximization" (which doesn't seem to be true at all).

(The example might seem deceptively simple in mathematical terms, but the difficulty in formalizing it in a utility functions framework is that you are comparing different possible worlds in which your opponent has different utility functions, and you are measuring inexploitability in terms of how much resources they take, not how their utility function is affected. So you at least need the concept of a "resource".)