Decision Theory and the Logic of Provability

Patrick LaVictoire

Machine Intelligence Research Institute

18 May 2015

Motivating Question

What decision problems remain non-obvious, even if we assume the agent is practically omniscient?

Philosophical example: Newcomb's Dilemma

In this talk, we will explore a very artificial context in which an agent has full knowledge of a decision problem and vast computing power to spend on it, and show that certain self-referential problems still emerge.

Framework for Decision Theory Problems

- Everything takes place within a virtual environment: a program $U()$ that takes no input.
- Within $U()$ there is a subprogram $A()$, which we will think of as the agent.
- $A()$ outputs an action, and $U()$ outputs a utility.
- We will usually assume that $U()$ and $A()$ are allowed to use arbitrary amounts of computation, up to and including access to halting oracles.
- $A()$ is allowed to know the source code of $U()$ and of $A()$. (Possible via quining.)

Example of Agent and Universe

```
def U():
  if A()=1:
    return 3
  else if A()=2:
    return 5
  else return 0
def A():
  return 2
```

Clearly, this agent returns action 2 and this universe returns utility 5.

Formalizing Our Questions

We can then generate some useful questions about decision theory in this context:

- Are there any counterintuitive outcomes for simple $A()$ and $U()$?
- Given a template for $U()$ where we can substitute in any $A()$, what $A()$ gets the best outcome?
- Are there algorithms $A()$ which reliably do well when substituted into many different universes $U()$?

Prisoner's Dilemma with Source Code Reading

We can consider multiplayer games with a universe $U()$ which calls on agents $A_1(), \ldots, A_N()$ and returns a vector of utilities $\langle u_1, \ldots, u_N \rangle$ representing the score for each agent. (Note: for each agent, the other agents can be seen as just part of the universe!)

Specifically, let's consider a one-shot Prisoner's Dilemma between two algorithms that get to read one anothers' source codes. (Variation on algorithmic tournaments of Iterated Prisoner's Dilemma!)

What algorithm might you submit to such a test?

Cliqueish Cooperation

We'll use $\ulcorner X \urcorner$ to represent the source code of an agent X, and $Y(\ulcorner X \urcorner)$ to represent the action of agent Y in the universe where it plays against agent X.

-
  ```
  def DefectBot(⌜X⌝):
      return D
  ```

-
  ```
  def CliqueBot(⌜X⌝):
      if ⌜X⌝ = ⌜CliqueBot⌝:
          return C
      else return D
  ```

Versions of CliqueBot independently invented by Howard (1988), McAfee (1984), Tennenholtz (2004).
But all of these would be in different cliques!

Circles of Mutual Cooperation

Goal: write an algorithm X that cooperates with Y only if that is necessary to imply that Y cooperates with X.

Problem: this is poorly defined and requires a good notion of counterfactual reasoning!

Simpler version: write an algorithm X that cooperates with Y if and only if Y cooperates with X. Let's call this idea FairBot.

FairBot

First attempt:

```
def EvalFairBot(⌜$X$⌝):
    if $X(\ulcorner EvalFairBot \urcorner) = C$:
      return $C$
    else return $D$
```

Problem: This never finishes evaluating against itself!

Slepnev (2011) considered seeking proofs in Peano Arithmetic:

```
def ProofFairBot(⌜$X$⌝):
    if PA$\vdash X(\ulcorner ProofFairBot \urcorner) = C$:
      return $C$
    else return $D$
```

Claim: This cooperates with itself with a finite proof search!

Löb's Theorem

Theorem (Löb's Theorem)

Let φ be a formula in the language of PA, and let $\Box\varphi$ be the formula in PA which asserts that φ is provable in PA. Then if $\Box\varphi \to \varphi$ is provable in PA, φ itself must be provable in PA.

- Example 1: φ a tautology like $2+2=4$ or $\top$. Then $\top$ is provable in PA, so is $\Box\top$, and so is $\Box\top \to \top$.
- Example 2: φ a contradiction like $2+2=5$ or $\bot$. $\bot$ is not provable, and neither is $\Box\bot$. $\Box\bot \to \bot$ is true in the standard model of $\mathbb{N}$ but not provable: it is equivalent to $\neg\Box\bot$ which asserts that PA is consistent.
- Example 3: φ is an undecidable statement like Gödel's G, where $G \leftrightarrow \neg\Box G$. Since then $\Box G \to G$ is equivalent to G itself, it cannot be provable in PA.

Löbian Cooperation

```
- def ProofFairBot(⌜$X$⌝):
    if PA$\vdash X(\ulcorner ProofFairBot \urcorner) = C$:
      return $C$
    else return $D$
```

Claim: This cooperates with itself with a finite proof search!

Proof of Claim: Let φ be the claim that the two copies of FairBot mutually cooperate. If φ is provable in PA, then each copy of FairBot will find the proof that the other copy cooperates, and this will cause each copy to actually cooperate. Thus we can formally show that $\Box\varphi \to \varphi$; thus φ is provable via Löb's Theorem.

Proof-Based Counterfactuals

This suggests that we can use provability in formal systems as a notion of decision-theoretic counterfactuals for algorithmic agents.

Explicitly, we will construct agents that seek proofs of statements $A() = a \to U() = u$ for various actions a and utilities u, and then take actions which achieve the best available utility.

This approach can fail for an interesting reason!

Spurious Proofs

```
• def U():
    if A()=1:
      return 5
    else return 3
• def A():
    utility[1], utility[2] $\leftarrow -\infty$
    if PA$\vdash A() = 2 \rightarrow U() = 3$:
      utility[2] $\leftarrow 3$
    for $u \in [5, 0]$:
      if PA$\vdash A() = 1 \rightarrow U() = u$:
        utility[1] $\leftarrow u$
    return argmax(utility)
```

Claim: $A() = 2$!

Proof of Claim

- Clearly PA$\vdash$ $A() = 2 \rightarrow U() = 3$, so utility[2] $\leftarrow$ 3.
- Clearly PA$\vdash$ $A() = 1 \rightarrow U() = 5$, so utility[1] $\leftarrow$ 5.
- I claim that PA$\vdash$ $A() = 1 \rightarrow U() = 0$. How? Löb's Theorem!
- Let φ be the statement $A() = 1 \rightarrow U() = 0$; we will prove that $\Box\varphi \rightarrow \varphi$, and thus that PA$\vdash$ φ.
- If $\Box\varphi$, then what actually happens? utility[1] $\leftarrow$ 0, and thus $A() = 2$. But that makes $A() = 1 \rightarrow U() = 0$ formally true.
- And this reasoning can be done in PA itself! Thus PA indeed proves that $\Box\varphi \rightarrow \varphi$, thus PA proves φ, and thus $A() = 2$.

Playing Chicken With Peano Arithmetic

The problem arose because PA could actually prove that $A() = 2$. We can prevent this by ensuring that PA cannot predict what $A()$ is, in the following way:

Before doing anything else, for each available action a, look for a proof of the statement $A() \neq a$; if such a check succeeds, immediately take action a.

Since we (from outside) presume PA consistent, we know that it cannot succeed at any such proof, and thus the value of $A()$ will be undecidable in PA.

Indeed, a chicken-playing decision theory works well, except for one problem: it cannot achieve Löbian cooperation with itself, since it cannot prove that another copy of it will cooperate.

Descending Order of Search

We assume that $A()$ chooses among a finite set $\mathcal{A}$ of actions, and $U()$ is known to lie in a finite set $\mathcal{U}$ of utilities.

```
• def A():
   for $u \in \mathcal{U}$, in descending order:
     for $a \in \mathcal{A}$:
       if PA$\vdash A() = a \to U() = u$:
         return $a$
   else return some default action
```

In a broad class of simple universes, we can show that this decision theory does as well as possible; moreover, it achieves mutual cooperation with itself on the Prisoner's Dilemma, and can be slightly modified to be inexploitable.

Logical Counterfactuals

A third response to spurious counterfactuals is to ask whether there is some formal criterion besides provability in which the counterfactual $A() = 1 \to U() = 5$ is valid but the counterfactual $A() = 1 \to U() = 0$ is not. If we can isolate such a criterion for logical counterfactuals, then we could directly use it in decision theory.

Intuitive example: "If Fermat's Last Theorem had been false, then elliptic curves would have had property X" is a valid logical counterfactual. "If Fermat's Last Theorem had been false, then 3 would be an even number" is not.

The Modal Logic GL

A nice tool for examining proof-based counterfactuals: Gödel-Löb modal logic. Briefly:

- Modal logic: propositional logic with the operator $\Box$ and new rules and axioms for handling it
- Modal logic GL: includes the Löbian axiom schema $\Box(\Box\varphi \to \varphi) \to \Box\varphi$
- Fixed point theorem: if agents and universe defined by formulas of modal logic, and agents depend on universe and each other only via provability, there's a unique fixed logical outcome and a polynomial-time algorithm for calculating it!
- Get a "model checker" for modal decision theories and modal game-playing agents.

Optimality for Modal Decision Theories

A universe is *extensional* if the outcome depends on the agent only via its action, i.e. if swapping the agent for another agent that outputs the same action will lead to the same outcome.

Theorem

For any provably extensional modal universe, if either the chicken-playing agent or the descending-search-order agent above does its deductions in PA+$\neg\Box^n\bot$ for n sufficiently large, it gets the best outcome achievable by any modal agent in that universe.

This is a strong condition on universes, and does not apply to universes (or other agents in those universes) that act differently based on the provability of the agent's action (e.g. depend on $\Box[A() = 1]$ rather than just on $A() = 1$). In fact, there is a pair of (non-extensional) modal universes such that no modal agent can be optimal on both.

Conclusions

- Proof-based counterfactuals are rich, interesting, and tractable for study.
- Proof-based counterfactuals can allow forms of safe mutual cooperation without prior coordination.
- Proof-based counterfactuals can include spurious counterfactuals that do not match our intuitions about logical counterfactuals.

Open Problems

- Are there restricted forms of optimality that hold for our modal decision theories in non-extensional universes?
- Are there tractable ways to study proof-based counterfactuals outside of the modal case?
- Is there a formal notion that corresponds to our intuitions about logical counterfactuals? (E.g. length of shortest proof, probability measures over models, etc)

Questions?

Thanks to CSER, MIRI, the organizers, the donors, my colleagues, and the counterfactual versions of myself whom I simulated in order to make decisions in the writing of this talk.