Robust Cooperation in the Prisoner's Dilemma: Program Equilibrium via Provability Logic

Patrick LaVictoire*, Mihaly Barasz, Paul Christiano, Benja Fallenstein, Marcello Herreshoff, and Eliezer Yudkowsky

May 31, 2013

Abstract

We consider the one-shot Prisoner's Dilemma between algorithms with access to one anothers' source codes, and apply the modal logic of provability to achieve a flexible and robust form of mutual cooperation. We discuss some variants, and point out obstacles to definitions of optimality.

1 Informal Introduction

Many philosophers have suggested that mutual knowledge of decision processes can enable rational cooperation in even the one-shot Prisoner's Dilemma. Rapoport [18] argued in the 1960s that two agents with mutual knowledge of each others' rationality should be able to cooperate. Howard [11] explains the argument thus:

> Nonetheless arguments have been made in favour of playing C even in a single play of the PD. The one that interests us relies heavily on the usual assumption that both players are completely rational and know everything there is to know about the situation. (So for instance, Row knows that Column is rational, and Column knows that he knows it, and so on.) It can then be argued by Row that Column is an individual very similar to himself and in the same situation as himself. Hence whatever he eventually decides to do, Column will necessarily do the same (just as two good students given the same sum to calculate will necessarily arrive at the same answer). Hence if Row chooses D, so will Column, and each will get 1. However if Row chooses C, so will Column, and each will then get 2. Hence Row should choose C.

*Partially supported by NSF Grant DMS-1201314.

Hofstadter [10] described this line of reasoning as "superrationality", and held that knowledge of similar cognitive aptitudes should be enough to establish it.

A promising advance in this discussion was the shift from qualitative philosophical arguments about counterfactuals and epistemic standpoints to a discussion of what could be achieved by computer programs (just as Axelrod's tournaments of iterated Prisoner's Dilemma strategies [3] shed light on the effectiveness of tough-but-fair strategies). What we lose in practicality by considering this more formal and artificial context, we gain in clarity, specificity and predictive power.

Relevantly, Binmore [4] considered game theory between Turing machines which had access to one anothers' Gödel numbers:

> ...a player needs to be able to cope with hypotheses about the reasoning processes of the opponents other than simply that which maintains that they are the same as his own. Any other view risks relegating rational players to the role of the "unlucky" Bridge expert who usually loses but explains that his play is "correct" and would have led to his winning if only the opponents had played "correctly". Crudely, rational behavior should include the capacity to exploit bad play by the opponents.
>
> In any case, if Turing machines are used to model the players, it is possible to suppose that the play of a game is prefixed by an exchange of the players' Gödel numbers.

(Binmore's analysis, however, eschews cooperation in the Prisoner's Dilemma as irrational!) In this context, one can of course include the usual one-shot strategies as Turing machines that return the same output regardless of the given input; we denote these algorithms as CooperateBot and DefectBot in order to distinguish them from the outputs Cooperate and Defect.

Howard [11] and McAfee [15] considered the formalizable special case of computer programs which take the opponent's source code before playing the Prisoner's Dilemma, and presented an example of an algorithm which would always return an answer, would cooperate if faced with itself, and would never cooperate when the opponent defected. (The solution discussed in both papers was a program that used quining of the source code to implement the algorithm "cooperate if and only if the opponent's source code is identical to mine"; we represent it in this paper as Algorithm 3, which we call CliqueBot on account of the fact that it cooperates only with the 'clique' of agents identical to itself.)

More recently, Tennenholtz [20] reproduced this result in the context of other research on multi-agent systems; such a Nash equilibrium (in terms of which program to submit) is called a "program equilibrium". This context led to several novel game-theoretic results, including folk theorems by Fortnow [8] and Kalai, Kalai, Lehrer and Samet [12], an answer by Monderer and Tennenholtz [16] to the problem of seeking strong equilibria (many-agent

Prisoner's Dilemmas in which mutual cooperation can be established in a manner that is safe from *coalitions* of defectors), a Bayesian framework by Peters and Szentes [17], and more.

However, these approaches have an undesirable property: not only does the process of writing such a program involve some arbitrary choices, but any two programs which make different choices cannot 'recognize' each other for mutual cooperation, never mind that they are functionally identical. (This problem can be patched somewhat, but not solved: it is impossible to write an algorithm that correctly verifies in general whether arbitrary algorithms are functionally identical to itself!) Thus mutual cooperation is inherently fragile for CliqueBots, and an ecology of such agents would be akin to an all-out war between incompatible cliques.

One attempt to put mutual cooperation on more general footing is the model-checking result of van der Hoek, Witteveen, and Wooldridge [9], which seeks "fixed points" of strategies that condition their actions on their opponents' output. However, in many interesting cases there are several fixed points, or none at all, and so this approach does not correspond to an algorithm as we would like.

Since the essence of this problem deals in counterfactuals—e.g. "what would they do if I did this"—it is worth considering modal logic, which was intended to capture reasoning about counterfactuals, and in particular the Gödel-Löb modal logic **GL** with provability as its modality. (See [5] and [13] for some good references on **GL**.) That is, if we consider provability in some particular formal system as a sufficient guarantee of validity, the structure of logical provability gives us a genuine framework for counterfactual reasoning, and in particular a powerful and surprising tool known as Löb's Theorem [14]:

Theorem 1.1. *Let S be a formal system which includes Peano Arithmetic. If ϕ is any well-formed formula in S, let $\Box\phi$ be the formula in a Gödel encoding of S which claims that there exists a proof of ϕ in S; then whenever $\mathsf{S} \vdash (\Box\phi \to \phi)$, in fact $\mathsf{S} \vdash \phi$.*

We shall see that Löb's Theorem enables a flexible and secure form of mutual cooperation in this context. In particular, we first consider the intuitively appealing strategy "cooperate if and only if I can prove that my opponent cooperates", which we call FairBot. If we trust the formal system used by FairBot, we can conclude that it is un-exploitable (in the sense that it never winds up with the sucker's payoff). When we play FairBot against itself (and give both agents sufficient power to find proofs), although either mutual cooperation or mutual defection seem philosophically consistent, it always finds mutual cooperation (Theorem 3.1)![1] Moreover, the underpinnings of this result (and the others in this paper) do not depend on the syntactical details of the programs, but only on their functional properties; therefore two such programs can cooperate, even if written differently.

That result suggests a focus on strategies that base their actions on the opponent's provable

[1]This result was proved by Vladimir Slepnev in an unpublished draft [19], and the proof is reproduced here with his permission.

behavior against other agents, rather than on purely syntactic features of the opponent's source code. After defining such a class of "modal agents", we can ask whether there exists a modal agent which improves on the main deficit of the above strategy: namely, that it fails to correctly defect against CooperateBot.[2] It turns out that indeed there is a modal agent that is unexploitable, cooperates mutually with itself and FairBot, and defects against CooperateBot; we call this agent PrudentBot.

After examining some variations on the concept of PrudentBot, we turn to the question of whether and in what sense it can be said to be optimal. Alas, there are several distinct obstacles to some natural attempts at a nontrivial and non-vacuous criterion for optimality among modal agents. This echoes the impossibility-of-optimality results of Anderlini [2] and Canning [6] on game theory for Turning machines with access to each others' source codes.

All the same, the results on Löbian cooperation represent a formalized version of robust mutual cooperation on the Prisoner's Dilemma, further validating some of the intuitions on "superrationality" and raising new questions on decision theory. The Prisoner's Dilemma with exchange of source code is analogous to Newcomb's problem, and indeed, this work was inspired by some of the philosophical alternatives to causal and evidential decision theory on that problem (see Drescher [7] and Altair [1]).

A brief outline of the structure of this paper: in Section 2, we introduce our formal framework and point out the equivalents of previous work on program equilibrium. In Section 3, we will introduce FairBot, prove that it achieves mutual cooperation with itself and cannot be exploited (Theorem 3.1); we then define the class of "modal agents", introduce PrudentBot, and show that it is also un-exploitable, cooperates mutually with itself and with FairBot, and defects against CooperateBot.

In Section 4, we show that a feature of PrudentBot—namely, that it checks its opponent's response against DefectBot—is essential to its functioning: modal agents which do not use third parties cannot achieve mutual cooperation with FairBot unless they also cooperate with CooperateBot. Then, in Section 5, we discuss possible criteria for an "optimal" modal agent, and show that none of these criteria are satisfiable by a modal agent. Finally, in Section 6, we will explain some of our aesthetic choices, and speculate on some future directions.

2 Formal Framework

There are two different formalisms which we will bear in mind throughout this paper. The first formalism is that of algorithms, where we can imagine two Turing machines `X` and `Y`, each of which is given as input the code for the other, and which have clearly defined outputs corresponding to the options C and D. (It is possible, of course, that one or both may fail to halt, though the algorithms that we will discuss will provably halt on all inputs.) This

[2]For a discussion of why this is the obviously correct response to CooperateBot, see Section 6.

formalism has the benefit of concreteness: we could actually program such agents, although the ones we shall deal with are often very far from efficient in their requirements. It has the drawback, however, that proofs about algorithms which call upon each other are generally difficult and untidy.

Therefore, we will do our proofs in another framework: that of logical provability in certain formal systems. More specifically, the agents we will be most interested in can be interpreted via modal formulas in Gödel-Löb provability logic, which is especially pleasant to work with. This bait-and-switch is justified by the fact that all of our tools do indeed have equivalently useful bounded versions; variants of Löb's Theorem for bounded proof lengths are well-known among logicians. The interested reader can therefore construct algorithmic versions of all logically defined agents in this paper, and with the right parameters all of our theorems will hold for such agents.

In particular, our "agents" will be formulas in Peano arithmetic, and our criterion for action will depend on the existence of a finite proof of its output in the tower of formal systems $\mathsf{PA}{+}\mathsf{n}$, where PA is Peano Arithmetic, and $\mathsf{PA}{+}(\mathsf{n}{+}1)$ is the formal system whose axioms are the axioms of $\mathsf{PA}{+}\mathsf{n}$, plus the axiom that $\mathsf{PA}{+}\mathsf{n}$ is consistent, i.e. that $\neg\Box\ldots\Box\bot$ with $n+1$ copies of $\Box$.

Fix a particular Gödel numbering scheme, and let `X` and `Y` each denote well-formed expressions with one free variable. Then let $X(Y)$ denote the formula where we replace each instance of the free variable in `X` with the Gödel number of `Y`. If such a formula is provable in some $\mathsf{PA}{+}\mathsf{n}$, we can interpret that as `X` cooperating with `Y`; if its negation is provable, we interpret that as `X` defecting against `Y`. Thus we can regard such formulas of arithmetic as decision-theoretic agents, and we will use "source code" to refer to their Gödel numbers.

Remark To maximize readability in the technical sections of this paper, we will use typewriter font for agents, which are formulas with a single free variable, like `X` and `CooperateBot`; we will use sans-serif font for the formal systems $\mathsf{PA}{+}\mathsf{n}$; and we will use italics for formulas with no free variables such as C, D, and $X(Y)$ for any agents `X` and `Y`. We will also use $\Box_n A$ to represent the formula which Gödel-encodes "there exists a proof of A in the formal system $\mathsf{PA}{+}\mathsf{n}$", and which can be expanded as the modal formula $\Box(\neg\Box\ldots\Box\bot \rightarrow A)$ with $n-1$ copies of $\Box$ on the inside.)

Of course, it is easy to create `X` and `Y` so that $X(Y)$ is an undecidable statement in all $\mathsf{PA}{+}\mathsf{n}$ (e.g. the statement that the formal system $\mathsf{PA}{+}\omega$ is consistent). But the philosophical phenomenon we're interested in can be achieved by agents which do not present this problem, and which in fact always return relative to an oracle for a specific $\mathsf{PA}{+}\mathsf{n}$.

Two agents which are clearly decidable and easy to define are the agent which always cooperates (which we will call `CooperateBot`, or `CB` for short) and the agent which always defects (which we will call `DefectBot`, or `DB`). In pseudocode:

Input : Source code of the agent `X`
Output: C or D
return C;

Algorithm 1: `CooperateBot (CB)`

Input : Source code of the agent `X`
Output: C or D
return D;

Algorithm 2: `DefectBot (DB)`

Remark In the Peano Arithmetic formalism, `CooperateBot` can be represented by a formula that is a tautology for every input, while `DefectBot` can be represented by the negation of such a formula. For any X, then, $\mathsf{PA} \vdash CB(X) = C$ and $\mathsf{PA} \vdash DB(X) = D$.

Note further that $\mathsf{PA} \nvdash \neg\Box DB(X) = C$, but that $\mathsf{PA}{+}1 \vdash \neg\Box DB(X) = C$; this distinction is essential.

Howard [11], McAfee [15] and Tennenholtz [20] introduced functionally identical agent schemas, which I've taken to calling `CliqueBot`; these agents use quining to recognize self-copies and mutually cooperate, while defecting against any other agent. In pseudocode:

Input : Source code of the agent `X`
Output: C or D
if `X`=`CliqueBot` **then**
  return C;
else
  return D;
end

Algorithm 3: `CliqueBot`

By the diagonal lemma, there exists a formula of Peano arithmetic which implements `CliqueBot`. (Quining in general isn't represented by a formula in Peano Arithmetic, but quining a particular formula, like one that checks for arithmetic equality of two variables, is permissible.)

`CliqueBot` has the nice property that it never experiences the sucker's payoff in the Prisoner's Dilemma. This is such a clearly important property that we will define it:

Definition We say that an agent `X` is *un-exploitable* if there is no agent `Y` such that $X(Y) = C$ and $Y(X) = D$.

However, `CliqueBot` has a notable drawback: it can only elicit mutual cooperation from agents that are identical to itself. Versions of `CliqueBot` that are logically equivalent but not syntactically identical will refuse to mutually cooperate, and patching this via identification of semantic equality is an intractable general problem. For this reason, it is worth looking for a more flexibly cooperative form of agent.

3 Modal Agents

A deceptively simple-seeming such agent is one we call `FairBot`. On a philosophical level, it cooperates with any agent that can be proven to cooperate with it. In pseudocode:

Input : Source code of the agent `X`
Output: C or D
if $\mathsf{PA} \vdash$ *X(FairBot)*$= C$ **then**
  return C;
else
  return D;
end

Algorithm 4: `FairBot (FB)`

`FairBot` references itself in its definition, but as with `CliqueBot`, this can be done via Kleene's recursion theorem. By inspection, we see that `FairBot` is un-exploitable: presuming that Peano arithmetic is sound, `FairBot` will not cooperate against any agent that defects against `FairBot`.

The interesting question is what happens when `FairBot` plays against itself: it intuitively seems plausible either that it would mutually cooperate or mutually defect. As it turns out, though, Löb's Theorem guarantees that since the `FairBot`s are each seeking proof of mutual cooperation, they both succeed and indeed cooperate with one another. (This was first shown by Vladimir Slepnev.)

Theorem 3.1. $\mathsf{PA} \vdash$ *FairBot(FairBot)*$= C$.

Proof (Simple Version): By inspection of `FairBot`, $\mathsf{PA} \vdash (\Box FB(FB) = C) \rightarrow FB(FB) = C$. By Löb's Theorem, Peano arithmetic does indeed prove that *FairBot(FairBot)*$=C$. □

However, it is a tidy logical accident that the two agents are the same; we will understand better the mechanics of mutual cooperation if we pretend in this case that we have two distinct agents, $\texttt{FairBot}_1$ and $\texttt{FairBot}_2$, and prove mutual cooperation without using the fact that their actions are identical.

Proof of Theorem 3.1 (Real Version): Let A be the formula "$FB_1(FB_2) = C$" and B be the formula "$FB_2(FB_1) = C$". By inspection, $\mathsf{PA} \vdash \Box A \rightarrow B$ and $\mathsf{PA} \vdash \Box B \rightarrow A$. This sort of "Löbian circle" works out as follows:

$$\begin{array}{cc}
\mathsf{PA} \vdash (\Box A \rightarrow B) \wedge (\Box B \rightarrow A) & \text{(see above)} \\
\mathsf{PA} \vdash (\Box A \wedge \Box B) \rightarrow (A \wedge B) & \text{(follows from above)} \\
\mathsf{PA} \vdash \Box(A \wedge B) \rightarrow (\Box A \wedge \Box B) & \text{(tautology)} \\
\mathsf{PA} \vdash \Box(A \wedge B) \rightarrow (A \wedge B) & \text{(previous lines)} \\
\mathsf{PA} \vdash A \wedge B & \text{(Löb's Theorem).}
\end{array}$$

□

Remark Unlike a `CliqueBot`, `FairBot` will find mutual cooperation even with versions of itself that are written in other programming languages. In fact, even the choice of formal system does not have to be identical for two versions of `FairBot` to achieve mutual cooperation! It is enough that there exist a formal system S in which Löbian statements are true, such that anything provable in S is provable in each of the formal systems used, and such that S can prove the above. (Note in particular that even *incompatible* formal systems can have this property: a version of `FairBot` which looks for proofs in the formal system $\mathsf{PA}+\neg\mathsf{Con}(\mathsf{PA})$ will still find mutual cooperation with a `FairBot` that looks for proofs in $\mathsf{PA}+1$.)

However, `FairBot` wastes utility by cooperating even with `CooperateBot`. (See the final section for the reasons we take this as a serious issue.) Thus we would like to know if there is a "best" agent in this context.

For many natural definitions of "best", there is clearly no such agent. For instance, there is no `X` such that for all `Y`, the utility achieved by `X` against `Y` is the highest achieved by any `Z` against `Y`. (To see this, consider `Y` defined so that $Y(Z) = C$ if and only if `Z`$\neq$`X`, or more generally, `Y` can "punish" `X` for any particular part of its source code which differs from that of `Z`.)

It is instructive to consider `FairBot` as a modal statement in Gödel-Löb provability logic. Namely, if we consider the actions of `FairBot` and any other agent `X` against one another, then the definition of `FairBot` is simply $A \leftrightarrow \Box B$, where A and B are the logical formulas $FB(X) = C$ and $X(FB) = C$, respectively.

In the case that the actions of `X` and `Y` can similarly be represented by logical formulas of the variables $A = [X(Y) = C]$ and $B = [Y(X) = C]$, such that $A \leftrightarrow \varphi_X(A, B)$ and $B \leftrightarrow \varphi_Y(A, B)$, where φ_X and φ_Y are each modal logic formulas using $\Box$, $\top$, and the logical operators $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$ and $\neg$, such that all instances of A and B within φ_X and φ_Y appear within expressions of the form $\Box\psi$, then there is a unique fixed point: A and B are each equivalent to constant sentences (modal logic formulas not using A or B). Since we trust the hierarchy of formal systems $\mathsf{PA}+\mathsf{n}$, we say $X(Y) = C$ whenever the constant formula is provable in some $\mathsf{PA}+\mathsf{n}$, and $X(Y) = D$ whenever the negation of the constant formula is provable in some $\mathsf{PA}+\mathsf{n}$. As all constant sentences are decidable in some $\mathsf{PA}+\mathsf{n}$, such agents are well-defined against one another. In fact, this can be extended to larger families of variables, and thereby allow `X` and `Y` to sample one anothers' actions against other third parties.

With this in mind, we define the class of modal agents:

Definition We say that `X` is a *modal agent* of rank $k \in \mathbb{N}$ if there is a fully modalized formula φ and a sequence of modal agents $\mathtt{Y}_1, \ldots, \mathtt{Y}_N$ each of rank $< k$ such that for any agent `Z`, if $P = [X(Z) = C]$, $Q = [Z(X) = C]$ and $R_i = [Z(Y_i) = C]$, then

$$P \leftrightarrow \varphi(P, Q, R_1, \ldots, R_N).$$

The reason for including "third parties" is that, as we shall see later, modal agents of rank 0 cannot discriminate between `FairBot` and `CooperateBot` without thereby losing `FairBot`'s cooperation!

Remark Using the properties of Kripke semantics, one can algorithmically derive the fixed-point solutions to the action of one modal agent against another; indeed, the results of this paper have additionally been checked by a computer program written by two of the authors.

Note that `CooperateBot`, `DefectBot` and `FairBot` are all modal agents, but `CliqueBot` cannot be (since any modal agent must treat identically two modal agents with logically equivalent formulas).

As it happens, there exists an un-exploitable modal agent which cooperates with `FairBot` and with itself, and defects against `DefectBot` and `CooperateBot`. (Note that we cannot simply use a variant of `CliqueBot` which cooperates with `FairBot` as well as with itself, and defects otherwise, since this is not a modal agent.) We call this agent `PrudentBot`, and define it as follows:

Input : Source code of the agent `X`
Output: C or D
if $\mathsf{PA} \vdash$ *X(PrudentBot)=C* and $\mathsf{PA{+}1} \vdash$ *X(DefectBot)=D* **then**
  return C;
end
return D;

Algorithm 5: `PrudentBot (PB)`

That is, the action of `PB` is given by the modal formula

$$[PB(X) = C] \leftrightarrow (\Box[X(PB) = C] \wedge \Box(\neg\Box\bot \rightarrow [X(DB) = D])).$$

Theorem 3.2. *`PrudentBot` is un-exploitable, mutually cooperates with itself and with `FairBot`, and defects against `CooperateBot`.*

Proof. Un-exploitability is again immediate from the definition of `PrudentBot` and the assumption that PA is consistent, since cooperation by `PrudentBot` requires a proof that its opponent cooperates against it.

In particular, $\mathsf{PA{+}1} \vdash PB(DB) = D$ (since $\mathsf{PA} \vdash DB(PB) = D$, $\mathsf{PA{+}1} \vdash \neg\Box[DB(PB) = C]$).

It is likewise clear that $\mathsf{PA{+}2} \vdash PB(CB) = D$.

Now since $\mathsf{PA{+}1} \vdash FB(DB) = D$, we again have the Löbian cycle where $[PB(FB) = C] \leftrightarrow \Box[FB(PB) = C]$, and of course vice versa; thus `PrudentBot` and `FairBot` mutually cooperate.

And as we have established $\mathsf{PA}{+}1 \vdash PB(DB) = D$, we have the same Löbian cycle for `PrudentBot` and itself. □

Remark It is important that we look for proofs of $X(DB) = D$ in a stronger formal system than we use for proving $X(PB) = C$; if we do otherwise, the resulting variant of PrudentBot would lose the ability to cooperate with itself. However, it is not necessary that the formal system used for $X(DB) = D$ be stronger by only one step than that used for $X(PB) = C$; if we use a much higher $\mathsf{PA}{+}\mathsf{n}$ there, we broaden the circle of potential cooperators without thereby sacrificing safety.

4 Third Parties

It feels a bit clunky, in some sense, for the definition of modal agents to include references to other, simpler modal agents. Could we not do just as well with a carefully constructed agent that makes no such outside calls (i.e. a modal agent of rank 0)?

In fact, we cannot, at least if we want our agent to mutually cooperate with `FairBot` and defect against `CooperateBot`.

Theorem 4.1. *Any modal agent* `X` *of rank 0 such that* $\mathsf{PA} \vdash [X(FB) = C]$ *must also have* $\mathsf{PA} \vdash [X(CB) = C]$.

Proof. We first reduce this to a statement about modalized formulas. The agent `X` is represented as a statement $P \leftrightarrow \varphi(P, Q)$; given any modal agent `Y`, we let $P = [X(Y) = C]$ and $Q = [Y(X) = C]$, and find the unique fixed point for P and Q (using the modal formula for Q derived from the definition of Y, and finitely recursing as needed.)

Now it is clear that the theorem corresponds to the following statement: for any modalized formula φ such that the fixed-point of $P \leftrightarrow \varphi(P, \Box P)$ is provable in PA, the fixed point of $P \leftrightarrow \varphi(P, \top)$ is also provable in PA.

We will prove this statement using Kripke semantics (see e.g. [13]). First, by the fixed-point theorem, there are constant sentences p and q logically equivalent to the unique fixed points of the two formulas above. (That is, p and q are built up from $\top$, the logical connectives, and $\Box$.)

Pick any GL Kripke model (K, R) such that $K \vDash p$ but $K \nvDash q$. (K, R) is a well-founded partial order, so there exists a minimal $w \in K$ with $w \nvDash q$. Since w is minimal with this property, $u \vDash q$ for all $u < w$.

Now note that for all $u < w$ in (K, R), we have $u \vDash p \leftrightarrow q$, and $u \vDash (\Box p) \leftrightarrow \top$. But since φ is fully modalized, the rules of Kripke semantics imply that $w \vDash \varphi(p, \Box p) \leftrightarrow \varphi(q, \top)$, which of course means $w \vDash p \leftrightarrow q$, which contradicts our assumptions. □

5 Obstacles to Optimality

We have seen that `PrudentBot` passes the most obvious decision-theoretic tests; could we hope that it might be optimal in some meaningful sense among the modal agents? As it happens, there are at least three different kinds of impediments to optimality among modal agents, which together make it very difficult to formulate any nontrivial and non-vacuous definition of optimality.

Most directly, for any modal agents `X` and `Y`, either their outputs are identical on all modal agents, or there exists a modal agent `Z` which cooperates against `X` but defects against `Y`. (For an enlightening example, consider the agent `TrollBot` which cooperates with `X` if and only if $\mathsf{PA} \vdash X(DB) = C$.) Thus any nontrivial and nonvacuous concept of optimality must be weaker than total dominance, and in particular it must accept that third parties could seek to "punish" an agent for succeeding in a particular matchup.

Secondly, for any modal agent `X` there exists $N \in \mathbb{N}$ such that $[X(Y) = C]$ is decidable in $\mathsf{PA}{+}\mathsf{N}$ for all modal agents `Y`. Therefore such an agent cannot simultaneously make the correct choices against `WaitFairBot`, which cooperates with `X` iff $\mathsf{PA}{+}\mathsf{N} \vdash \bot$ XOR $\Box_N[X(WFB) = C]$, and against `WaitDefectBot`, which cooperates with `X` iff $\mathsf{PA}{+}\mathsf{N} \vdash \bot$. (Note that all predictions that a modal agent can make about these agents, in $\mathsf{PA}{+}\mathsf{N}$ or lower, are identical.)

And there is a third issue illustrated by the following agent:

Input : Source code of the agent `X`
Output: C or D
if $\mathsf{PA} \vdash$ *X(FairBot)*$= C$ **then**
  return C;
else
  return D;
end

Algorithm 6: `JustBot (JB)`

That is, `JustBot` cooperates with `X` if and only if `X` cooperates with `FairBot`. Clearly, `JustBot` is exploitable by some algorithm (in particular, consider the algorithm which cooperates only with the corresponding `FairBot` and with nothing else), but surprisingly it is not exploitable by any modal agent:

Theorem 5.1. *There does not exist a modal agent that cooperates against* ***`FairBot`*** *but defects against* ***`JustBot`****.*

Proof. Suppose there exist such agents; we can then take `X` of minimal modal rank such that $X(FB) = C$ and $X(JB) = D$. Note that for all $\mathtt{Z}_i$ referenced in the formula for `X`, we must have $Z_i(FB) = Z_i(JB)$ by minimality, and obviously $FB(Z_i) = JB(Z_i)$.

Thus if `X` has the formula $P \leftrightarrow \varphi(P, Q, R_1, \ldots, R_N)$, we see that the fixed points $r_{i,FB}$

and $r_{i,JB}$ must all be logically equivalent. Thus there exists a formula $\tilde{\varphi}$ such that $p_{FB} \leftrightarrow \tilde{\varphi}(p_{FB}, \Box p_{FB})$ and $p_{JB} \leftrightarrow \tilde{\varphi}(p_{JB}, \Box p_{FB})$.

Now the proof resembles the proof of Theorem 4.1; we consider any Kripke model in which $K \vDash p_{FB}$ but $K \nvDash p_{JB}$, and note that if $w \in K$ is minimal such that $w \nvDash p_{JB}$, then for any $u < w$, $u \vDash p_{FB} \leftrightarrow p_{JB}$; so by the fact that $\tilde{\varphi}$ is still modalized, $w \vDash \varphi(p_{FB}, \Box p_{FB}) \leftrightarrow \tilde{\varphi}(p_{JB}, \Box p_{FB})$ so that $w \vDash p_{FB} \leftrightarrow p_{JB}$ after all, which contradicts our assumptions. □

Despite these reasons for pessimism, we have not actually ruled out the existence of a nontrivial and non-vacuous optimality criterion which corresponds to our philosophical intuitions about "correct" decisions. Additionally, there are a number of ways to depart only mildly from the modal framework (such as allowing quantifiers over agents), and these can invalidate some of the above obstacles.

6 Fair or Prudent?

One might ask (on a philosophical level) why we object to FairBot in the first place; isn't it a feature, not a bug, that this agent offers up cooperation even to agents that blindly trust it rather than defecting against them? We respond that it is too tempting to anthropomorphize agents in this context, and that many problems which can be interpreted as playing a Prisoner's Dilemma against a CooperateBot are situations in which one would not hesitate to "defect" in real life without qualms.

For instance, consider the following situation: You discover that your home has an ant infestation, and are about to call the exterminator when you realize that you're locked in a prisoner's dilemma with the ants. You can choose whether to bring in the exterminator, and likewise, the ants are capable of swarming your person and making you flee the house, thus leaving them the sandwich you just made (which would more than compensate the nearby colonies for the worker ants lost in such an assault). On the other hand, you can predict that the ant colonies will not in fact swarm you, as that level of strategic behavior against humans is beyond their capacity (and swarming a larger animal is usually not worthwhile for most ant species). Therefore, the ant colonies are in the position of a CooperateBot. Would you really refrain from calling an exterminator as a *quid pro quo* for their expected "cooperation"?

The example is artificial, but any division of the world into agents has the feature that one can define as a CooperateBot any agent incapable of "defecting" in some way that would actually conduce toward its utility function. And most of us feel no compunction about optimizing our human lives without worrying about that of the ant colony. (Note that it would, in fact, be different if an ant colony were intelligent enough, not only to mount strategic assaults, but to determine whether a human being is calling an exterminator and base their actions on that! In such a case, one might well negotiate with the ants. Alternatively, if one's concern for the well-being of ants or ant colonies reached a comparable level

to one's concern for an ant-free home, that would change the payoff matrix.)

In a certain sense, PrudentBot is actually "good enough" among modal agents that one might expect to encounter: there are bound to be agents (CooperateBot and DefectBot) whose action fails to depend in any sense upon their opponent, and other agents (FairBot, PrudentBot, etc) whose action depends on their opponent's action in a sensible way. One should not expect to encounter a TrollBot, WaitBot or JustBot arising naturally! But it's worth pondering if this reasoning can be made formal in any elegant way.

Does this, then, justify cooperation in a real-life Prisoner's Dilemma among sufficiently intelligent and rational agents? In a word, no, not yet.[3] We human beings don't have easily readable source codes, nor would we be able to prove much if we did; and our abilities to read each other psychologically, while occasionally quite impressive, bear only the slightest analogy to our extremely artificial setup. Governments and corporations may be closer analogues to our agents (and indeed, game theory has been applied much more successfully on that scale than on the human scale), but the authors would not consider the application of these results to such organizations to be straightforward, either. The theorems herein are simply a demonstration that a more advanced approach to decision theory (i.e. one which does not fail on what we consider to be commonsense problems) is possible, not yet a demonstration that it is practical. The authors have a fair bit of hope that it more practical results in this vein can be developed, in addition to yet more impressive theoretical results.

Acknowledgments

This project grew out of a philosophy discussion on a group blog and mailing list, and several results were proved at an April 2013 workshop run by the Machine Intelligence Research Institute. Thanks to everyone who took part in this discussion, in particular Alex Altair, Stuart Armstrong, Andrew Critch, Wei Dai, Daniel Dewey, Gary Drescher, Bill Hibbard, Vladimir Nesov, Vladimir Slepnev, Jacob Steinhardt, Nisan Stiennon, Jacob Taylor, and Qiaochu Yuan.

References

[1] Alex Altair. A comparison of decision algorithms on newcomblike problems.

[2] Luca Anderlini. Some notes on church's thesis and the theory of games. *Theory and Decision*, 29(1):19–52, 1990.

[3] However, to borrow what Randall Munroe said about correlation and causation, this form of program equilibrium does waggle its eyebrows suggestively and gesture furtively (toward cooperation in the Prisoner's Dilemma) while mouthing 'look over there'.

[3] R. Axelrod and W. D. Hamilton. The Evolution of Cooperation. *Science*, 211:1390–1396, March 1981.

[4] Ken Binmore. Modeling rational players: Part i. *Economics and Philosophy*, 3(02):179–, 1987.

[5] G. Boolos. *The Logic of Provability*. Cambridge University Press, 1995.

[6] David Canning. Rationality, computability, and nash equilibrium. *Econometrica*, 60(4):877–88, July 1992.

[7] G.L. Drescher. *Good And Real: Demystifying Paradoxes from Physics to Ethics*. A Bradford Book. Mit Press, 2006.

[8] Lance Fortnow. Program equilibria and discounted computation time. *Proc. 12th Conference on Theoretical Aspects of Rationality and Knowledge*, pages 128–133, 2009.

[9] Wiebe Hoek, Cees Witteveen, and Michael Wooldridge. Program equilibrium—a program reasoning approach. *International Journal of Game Theory*, pages 1–33, 2011.

[10] Douglas R. Hofstadter. *Metamagical Themas: Questing for the Essence of Mind and Pattern*. BasicBooks, 1985.

[11] J.V. Howard. Cooperation in the prisoner's dilemma. *Theory and Decision*, 24(3):203–213, 1988.

[12] Adam Tauman Kalai, Ehud Kalai, Ehud Lehrer, and Dov Samet. A commitment folk theorem. *Games and Economic Behavior*, 69(1):127 – 137, 2010. Special Issue In Honor of Robert Aumann.

[13] Per Lindström. Provability logic—a short introduction. *Theoria*, 62(1-2):19–61, 1996.

[14] M. H. Löb. Solution of a Problem of Leon Henkin. *The Journal of Symbolic Logic*, 20(2):pp. 115–118, 1955.

[15] R Preston McAfee. Effective computability in economic decisions.

[16] Dov Monderer and Moshe Tennenholtz. Strong mediated equilibrium. *Artif. Intell.*, 173(1):180–195, January 2009.

[17] Michael Peters and Balázs Szentes. Definable and contractible contracts. *Econometrica*, 80(1):363–411, 2012.

[18] A. Rapoport. *Two-Person Game Theory*. Dover Books on Mathematics Series. Dover, 1999.

[19] Vladimir Slepnev. Self-referential algorithms for cooperation in one-shot games (unpublished draft).

[20] Moshe Tennenholtz. Program equilibrium. *Games Econom. Behav.*, 49(2):363–373, 2004.