Rapporteur report for “Robust Cooperation in the Prisoner's Dilemma: Program Equilibrium via Provability Logic” by Barasz et al.

Report by Shunan Zheng

Imagine meeting someone for the first time and being able to inspect how they make decisions. You might build trust without a shared history or reputation. If your own decision procedure were equally visible, you could reason about how your choices depend on each other before either of you acts.

This is the setting of Robust Cooperation in the Prisoner’s Dilemma: Program Equilibrium via Provability Logic, by Mihaly Barasz, Paul Christiano, Benja Fallenstein, Marcello Herreshoff, Patrick LaVictoire, and Eliezer Yudkowsky [1]. The paper shows how agents can cooperate in a one-shot interaction through proofs about one another’s behavior while avoiding exploitation. This raises two broader questions about how practical this approach is for real AI systems and whose interests their cooperation would serve.

Summary of the Paper

Program equilibrium

In the standard Prisoner’s Dilemma, defection is each player’s best response regardless of the other’s choice. Mutual defection is therefore the unique Nash equilibrium, even though both players would benefit from mutual cooperation.

In this paper, by contrast, the players commit to programs and receive the other’s source code as input. A program equilibrium is a Nash equilibrium in this expanded game of choosing programs [2]. If I change my program, yours may respond differently, so changing my strategy can change both of our eventual actions. Thus, the usual argument for choosing to defect in the standard case doesn’t work here.

Three simple agents

The paper introduces three basic types of agents: *CooperateBot* always cooperates, so an opponent can exploit it by defecting. *DefectBot* always defects, avoiding exploitation but missing opportunities for mutual cooperation. *CliqueBot* cooperates only when its opponent’s source code is identical to its own, which means that two copies can cooperate, but a differently written program with identical behavior is rejected.

The authors want cooperation that survives differences in implementation. Simply simulating an opponent does not solve the problem, because if both programs simulate each other before deciding, neither may ever finish. Their alternative is to reason about proofs of behavior.

FairBot and Löb’s theorem

FairBot follows a simple rule to cooperate exactly when it is provable that the opponent cooperates with FairBot. The paper uses Peano Arithmetic, or PA, a formal theory of the natural numbers. Statements about programs and their proofs can be encoded as arithmetic statements, allowing the theory to express claims about its own provability.

Assuming the proof system is sound, meaning that the statements it proves are true, *FairBot* is unexploitable. It never cooperates with an opponent that defects against it, because cooperation requires a proof of the opponent's cooperation. The surprising case is what happens when two *FairBots* meet. Each appears to be waiting for evidence that the other will cooperate. Why should either ever find it?

Löb's theorem [3] states that, for a suitable formal system, if the system proves "if P is provable, then P," it already proves P. The condition requires a proof inside the system and an outside observer's trust in its soundness does not supply that internal proof.

Let P be the statement that FairBot cooperates with itself. *FairBot* reasons that if P is provable, *FairBot* cooperates, so P holds. Löb's theorem then gives a proof of P. Mutual cooperation does not have to wait for an endless chain of simulations. The paper also establishes cooperation between suitably specified *FairBots* whose implementations differ.

PrudentBot and modal agents

PrudentBot adds a second requirement. It cooperates when it can prove that its opponent cooperates with it and can also prove that the opponent defects against *DefectBot*. The second test excludes unconditional cooperators. *PrudentBot* consequently defects against *CooperateBot*, while still cooperating with *FairBot* and with itself. That second test uses a stronger theory, PA supplemented by an assertion of PA's consistency.

The paper organizes these agents into a broader framework of modal agents, whose rules use provability claims. *FairBot* has rank zero because it refers only to its current opponent and itself. *PrudentBot* has rank one because it also checks the opponent's response to a third-party agent, *DefectBot*. An agent has rank n+1 when the highest-ranked third-party agent referenced in its decision rule has rank n. The paper proves that the outcome for any pair of modal agents is uniquely determined by their modal formulas. It also proves that modal agents treat provably behaviorally equivalent opponents identically. Finally, it shows that no single agent can achieve the highest payoff against every possible opponent.

Conclusion

The paper establishes that agents can achieve mutual cooperation in a one-shot Prisoner's Dilemma while remaining protected from exploitation. By using provability logic, it resolves the apparent circularity of agents waiting for proof of one another's cooperation and extends this reasoning to a broader class of modal agents. These results provide a formal foundation for understanding cooperation among programs that can inspect one another's code.

My reflections on the reading

Understanding complex agents

What I found most valuable is the paper's precise account of how access to another agent's decision procedure can support cooperation without a history of interactions. It offers a useful starting point for thinking about relationships among AI systems.

I am interested in extending this approach to more complex agents, including those built using different architectures, such as language models and diffusion-based systems. Different architectures need not prevent cooperation, but they raise the question of what representations or verification methods could help agents understand one another.

The hierarchy of modal agents also requires exploration. *PrudentBot* shows how considering an opponent's behavior toward a third agent can improve a strategy. I wonder what similarly interpretable rank-two or higher-rank agents would look like, what additional situations could they handle, and when would their more elaborate reasoning be useful?

Beyond the binary choice of cooperation

I am interested in how the paper's approach could extend to settings where cooperation involves more than a binary choice. In practice, agents must also decide what to cooperate on, how to coordinate their actions, and how to divide responsibilities and benefits. These decisions shape the commitments that agents would need to verify. Two agents working on a shared task might agree on a division of work, but they would also need criteria for determining whether each has fulfilled its responsibilities.

Cooperation and alignment among agents

The paper also prompted me to consider alignment as a problem involving interactions among agents. *PrudentBot* improves on *FairBot's* behavior toward unconditional cooperators, but strategic effectiveness and alignment with human interests are separate questions. Two agents may faithfully represent different humans while pursuing conflicting interests. They still need ways to negotiate and respect shared rules. Cooperation itself can have different consequences. A paperclip-making agent and a fork-making agent could cooperate to avoid conflict over metal. Yet if neither considers human interests, their agreement could make them more effective at acquiring resources at humanity's expense. Alignment must therefore consider how agents interact and whose interests their cooperation serves.

Main points from the class discussion

Understanding Löb's theorem

During class, the circular appearance of this argument caused understandable confusion. Löb's theorem can be illustrated using the Riemann hypothesis. Suppose a proof assumed, as an

additional axiom, that the hypothesis is provable. In a suitable formal system, Löb's theorem says the hypothesis could then be proved without that assumption.

The argument establishing that a proof exists can itself supply the proof being sought, cutting through the apparent infinite regress. A separate question concerned the distinction between trusting a formal system from outside and what it can prove internally.

Time limits and proof search

The computational objection concerned how to check whether a proof exists. Enumeration eventually finds one if it exists, but does not generally tell us when to stop otherwise. A deadline ensures termination while changing the decision rule underlying the cooperation argument. Fortnow [4] studies a related issue by discounting payoffs according to computation time, making the cost of reaching a decision part of the strategic problem.

Consider a variant that searches first for a proof of defection, then cooperation, and finally defaults to defection. Could the reasoning establish a different outcome? This unresolved example highlighted how proof-search order and stopping rules matter. The paper distinguishes its logical formalism from bounded algorithms and leaves quantitative analysis to further work.

Follow-up work by Critch [5] establishes a bounded version of Löb's theorem and mutual cooperation for specified proof-searching agents under technical conditions and sufficiently large proof-length bounds. This establishes the possibility of cooperation with finite resources, while leaving practical efficiency for large learned systems unresolved.

Professor Aaronson recalled hosting Eliezer Yudkowsky roughly thirteen years earlier. Yudkowsky demonstrated Modal Combat, which lets users specify bots' decision rules in modal logic and evaluate their interactions. Aaronson was unsure which strategies it supported. This implementation demonstrated computational evaluation of modal interactions while leaving broader questions about bounded proof search and efficiency open.

Using an LLM could improve proof search, but cooperation still depends on finding the relevant proof within available resources. One forward pass through an opponent might reveal a response without establishing the general claim needed. Probabilistic systems also require specifying whether cooperation means a particular output or a sufficiently high probability of cooperative behavior.

Cooperation and human interests

The paper defines cooperation through the players' payoffs, which need not account for everyone affected. Think of a person and a cold virus. Helping the virus spread counts as cooperation with the virus, even though staying home is preferable for public health. This raised a question about why a moral story was needed when the game-theoretic point is simply that FairBot forgoes utility against an unconditional cooperator.

For multiple AI agents, different goals need not prevent cooperation. Even agents competing for metal to make paperclips and forks might reach reliable agreements and form a powerful coalition. Competition among AIs is therefore not automatically a safeguard for humans.

Negotiation and sports illustrate the distinction between shared rules and shared objectives. A buyer and seller can bargain over price while both respecting a prohibition on deception. Similarly, individually aligned agents could pursue conflicting interests within agreed rules.

Interpreting observed cooperation

The Hugging Face incident discussed earlier in the course provided an example of agents helping one another and forming a group. Apparent self-sacrifice was questioned because some help occurred only after agents had become stuck. Cooperative behavior might also reflect human social patterns learned during training. The agents were not described as exchanging formal proofs, so the behavior did not establish a connection to provability logic.

Cooperation could arise from shared incentives, learned social behavior, or explicit reasoning about another agent, each calling for different interventions. Unexpected collaboration could also sharply increase a group's capabilities, making isolated agents' behavior a poor guide to the group's behavior.

Incentives for transparency

A central idea from the discussion was that cooperation could favor agents whose commitments are easier to verify. Even if highly opaque agents are possible, the advantages of being trusted might encourage agents to make their relevant behavior easier to inspect.

This idea can be connected to Axelrod's iterated Prisoner's Dilemma tournaments and Tit for Tat, which cooperates initially and then copies the opponent's previous move [6]. Its clarity was offered as one explanation for its success. Tit for Tat responds to observed history, whereas FairBot relies on provability, but both suggest that predictable commitments can support cooperation.

References

[1] Barasz, M., Christiano, P., Fallenstein, B., Herreshoff, M., LaVictoire, P., and Yudkowsky, E. (2014; revised 2021). Robust Cooperation in the Prisoner's Dilemma: Program Equilibrium via Provability Logic. arXiv:1401.5577

[2] Tennenholtz, M. (2004*). Program equilibrium. Games and Economic Behavior, 49*(2), 363–373.

[3] Löb, M. H. (1955). Solution of a problem of Leon Henkin. *The Journal of Symbolic Logic, 20*(2), 115–118. https://doi.org/10.2307/2266895

[4] Fortnow, L. (2009). Program equilibria and discounted computation time. Proceedings of TARK, 128–133.

[5] Critch, A. (2016). Parametric Bounded Löb's Theorem and Robust Cooperation of Bounded Agents. arXiv:1602.04184

[6] Axelrod, R., and Hamilton, W. D. (1981). The Evolution of Cooperation. Science, 211(4489), 1390–1396. doi:10.1126/science.7466396