COOPERATION IS PROVABLY REQUIRED IN A VERSION OF THE NOISY ITERATED PRISONER'S DILEMMA

ARVID LUNNEMARK

1. Introduction

The prisoner's dilemma is a classic symmetric two-player game, where both players acting in their own best interests leads to an outcome that is not optimal for either of them. There are two actions, cooperate (C) and defect (D), with the payoffs represented by the matrix

$$\begin{array}{ccc} & C & D \\ C & (R,R) & (S,T) \\ D & (T,S) & (P,P) \end{array}$$

where $T > R > P > S$ (the row player gets the first in each pair of numbers). Given the other player's action, a player always maximizes their payoff by choosing D, but with both doing so, the players end up in the (D,D) state, which is worse for both than the (C,C) state since $R > P$. The game can been used to model many real-world situations, including countries failing to act to stop climate change, doping in sport, and economic competition.

copy the table from axelrod's book because it's much better and much clearer!!!

To make the game more interesting, one can consider playing it multiple times in a row with multiple players. One might loosely connect the repeated multiplayer game to evolution: why have humans evolved to cooperate with each other? In a seminal paper in 1980, Axelrod presented evidence that in a repeated setting with multiple players facing off in a tournament, cooperation can arise as the strategy of choice even for a selfish player [1]. In particular, the mostly cooperating Tit-for-tat strategy, depicted in fig. 1, won his tournament. The idea is that even though Tit-for-tat loses when played against defecting strategies, that is outweighed by it being heavily rewarded when cooperating with other cooperating strategies.

Clearly, however, the winning strategy in a tournament depends on the composition of strategies in that tournament. If all strategies had been of the type to always defect, Tit-for-tat would not have won the Axelrod tournament. Therefore, follow-up tournaments and simulations have been run, and perhaps surprisingly, most provide further evidence that cooperation is the prevailing strategy, as Axelrod summarizes in a 1981 book [2]. There, he also introduces an evolutionary model that can be used for analyzing the game theoretically: strategies are considered to exist in a population that

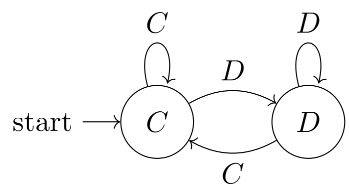

Figure 1. The Tit-for-tat strategy. At any one point, it is in one of two states, taking the action corresponding to the label of the state. Upon perceiving its opponent's move, it decides to switch state if the opponent does something different.

evolves over time in an evolutionary way (more successful strategies reproduce, and mutations can happen). In that context, a strategy is said to be evolutionarily stable if, supposing it controls a large share of the population, it resists being overtaken by any other strategy entering the population in small numbers. That is, populations of evolutionarily stable strategies form the equilibruim states of the evolutionary process, and thus, one may say that evolutionarily stable strategies are the "best" strategies.

In light of Axelrod's tournaments highlighting the effectiveness of cooperation, it has long been a goal to prove that a cooperating strategy like Tit-for-tat is evolutionarily stable. A few variants of this has been shown: First, Nowak has shown that in the finitely repeated game, a strategy that always defects is sometimes not evolutionarily stable [6]. Second, Binmore has shown that in the infinitely repeated game where strategies are modeled as finite automatons where having more states comes at a cost, a strategy needs to cooperate with itself to be stable [3]. And third, Fundenberg and Maskin have shown that a strategy needs to cooperate with itself to be stable also in the infinitely repeated game in the presence of a certain notion of infinitesimally small noise [4].

Notably, all these results impose additional restrictions on the setup, and as noted by Fundenberg and Maskin, one has to do that: in the deterministic infinitely repeated game, one could create a strategy that can self-identify — if the opponent ever deviates from the pattern, the strategy resorts to defecting for the rest of the game. This strategy will be evolutionarily stable, but is not cooperative.

In this paper, we, like Fundenberg and Maskin, choose the addition of noise as our restriction. In contrast to them, however, we present a model where, at every step, each strategy has a tiny probability p of doing the wrong thing, which is arguably the most natural way of modeling noise. In section 2, we define the setup of our version of the problem in detail. Then, in section 3, we state the two main results: that a strategy needs to be cooperative to be evolutionarily stable, and that such evolutionarily stable strategies exist. In section 4, we prove our results. Finally, in section 5 we briefly discuss potential directions for future work.

2. Setup

In this section we define our setup. In summary, we consider the infinitely repeated prisoner's dilemma played by finite automata in infinite populations, evolving evolutionarily in the presence of noise.

2.1. Formal Definition.

First, we define the reward function.

Definition 2.1. The *prisoner's dilemma* is a symmetric two-player game with two actions, cooperate (C) and defect (D), where, if player 1 selects action a and player 2 selects action b, player 1 gets the reward

$$r(a,b) = \begin{cases} R & \text{if } a = C, b = C \\ T & \text{if } a = D, b = C \\ S & \text{if } a = C, b = D \\ P & \text{if } a = D, b = D \end{cases}$$

We require $T > R > P > S$ and $2R > T + S$.

When we study the *iterated* prisoner's dilemma, we want to look at strategies that determine their next move based on the history of previous moves. We restrict ourselves to strategies that can be implemented on a computer with finite memory.

Definition 2.2. A *strategy* s is a Moore machine (finite automaton with outputs) over the input and output alphabet $\{C, D\}$.

We will consider strategies in the presence of noise. To model that, we will assume that a strategy has a probability $1 - p$ of following the correct transition and a probability p of following the incorrect transition, at every step. Note that this models noise in *perception*. One could also imagine modeling noise in *action taken*, but it is easy to see that the two are equivalent up to a change in the values of R, T, S and P.

We can now begin to define the outcome of two strategies playing against each other in the infinitely repeated game. We do this using Markov chains.

Definition 2.3. Suppose that strategy s_1 plays against strategy s_2. This defines an s_1-s_2 *Markov chain* where each state is a tuple (c_1, c_2) with c_1 being a state in s_1 and c_2 a state in s_2. The transition probabilities are defined in the obvious way:

$$p_{(c_1,c_2)\to(c_1',c_2')} = p_{c_1\to c_1'} \cdot p_{c_2\to c_2'},$$

where $p_{c_1\to c_1'}$ is 1 if s_1 always transitions from c_1 to c_1', 0 if it never does, $1 - p$ if the output of s_2 at state c_2 causes s_1 to transition from c_1 to c_1', and p if the opposite of that output causes the transition; $p_{c_2\to c_2'}$ is defined similarly.

Markov chains naturally lead themselves to the study of limiting cases.

Definition 2.4. Let a_0 be the start state of s_1 and b_0 the start state of s_2. The *time average distribution* of the s_1-s_2 Markov chain, denoted π, is the distribution such that

$$\pi_{c_1,c_2} = E\left[\text{fraction of time in state } (c_1, c_2) \mid \text{start in state } (a_0, b_0)\right]$$

where the expected fraction of time is taken over the infinite sequence of states representing a random walk.

Note that the time average distribution given a start state always exists, regardless of the structure of the Markov chain. Now, we can define the payoff.

Definition 2.5. Let $r(c_1, c_2)$ refer to the reward that s_1 gets when s_1 takes the action in state c_1 and s_2 takes the action in state c_2. Then, the *payoff that s_1 gets when playing against s_2* is

$$v_{s_1}(s_2) = \sum_{(c_1,c_2)} \pi_{c_1,c_2} \cdot r(c_1, c_2),$$

where the sum is taken over all states (c_1, c_2) in the s_1-s_2 Markov chain.

For notational convenience, we may also make $r(c_1, c_2)$ into a vector, denoted by r, and write this as the dot product

$$v_{s_1}(s_2) = \pi \cdot r.$$

That is, the payoff that s_1 gets when playing against s_2 is simply an average of the reward it gets in each possible state of the Markov chain weighted by the fraction of time that is spent there.

We're now ready to look at how strategies interact.

Definition 2.6. A *population* of strategies $P = (S, f)$ is a set S of strategies and a function $f : S \to (0, 1]$ such that $\sum_{s \in S} f(s) = 1$, representing the frequency of each strategy in the population.

Definition 2.7. The *fitness* of a strategy s in a population $P = (S, f)$ is

$$F(s) = \sum_{s' \in S} f(s') v_s(s').$$

One can think of this as saying that we have infinitely many members of the population, interacting with each other evenly, and that the fitness of a strategy is its expected payoff. Having infinitely many interactions like this justifies the usage of expectation when definining $v_{s_1}(s_2)$.

We can now use the fitness of a strategy to compare it with other strategies in the same population. If a strategy s_1 has a higher fitness than another strategy s_2, we say that the frequency of s_1 will increase at the expense of the frequency of s_2, in the next step of the evolutionary process. This is getting us close to how we want to define stable strategies; our next move is looking not only at a single evolutionary step, but the entire evolutionary process.

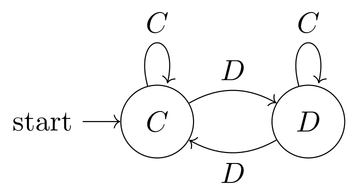

FIGURE 2. The Pavlov strategy. Similar to but not the same as the Tit-for-tat strategy.

Definition 2.8. A strategy s_1 is *ϵ-invadable* if there exists a strategy s_2 such that in all populations P with $S = \{s_1, s_2\}$ and $f(s_2) \geq \epsilon$, we have

$$F(s_2) > F(s_1)$$

That is, if s_1 is ϵ-invadable, there exists a strategy s_2 that can start as only a tiny fraction ϵ of the total population, and consistently have higher fitness than s_1, eventually causing s_2 to overtake and then eliminate s_1 completely. We are now finally ready to define evolutionary stability.

Definition 2.9. A strategy s_1 is *evolutionarily stable* if there exists parameters p_0 and ϵ_0 with $0 < p_0, \epsilon_0 < 1$, such that for all $p < p_0$, and all $\epsilon < \epsilon_0$, s_1 is not ϵ-invadable.

That is, a strategy s_1 is evolutionarily stable if, as the noise probability p goes to 0, it can withstand invasion attempts from any strategy that starts off as a tiny fraction of the population.

separate this definition out into a pure evolutionarily stable definition, and then an evolutionarily stable for infinitesimally small probabilities definition?? would make interpretation much clearer.

2.2. **Interpretation.** Having concluded our formal definition of evolutionary stability, it might be helpful to see examples of concrete setups in which this definition makes sense.

Consider the following world.

WRITE THIS SECTION?? I'M NOT SURE THAT MY MODEL MAKES MUCH SENSE ANYMORE THOUGH, BECAUSE IT REQUIRES PEOPLE TO PLAY AGAINST THEMSELVES WHICH IS A NONO. also the probability go-

3. RESULTS

We can now state our results! Together, the following theorem and conjecture imply that in the setup described here, mutual cooperation arises as the only stable choice.

Theorem 3.1. Suppose that a strategy s_1 is evolutionarily stable. Then $\lim_{p \to 0} v_{s_1}(s_1) = R$.

Conjecture 3.2. Suppose that $2R > T + P$. Then, the Pavlov strategy, depicted in fig. 2, is evolutionarily stable.

Remark. Tit-for-tat, depicted in fig. 1, is not evolutionarily stable. When played against itself, it ends up spending just as much time in the defection state as in the cooperation state, because as soon as one mistake is made, it goes into a mutual defection cycle with its clone that is only broken by an additional mistake. Thus, $v_s(s)$ where s is Tit-for-tat is much smaller than R, which means that it is not evolutionarily stable by theorem 3.1. Notably, this shows that our results are different, and perhaps stronger, than Fudenberg and Maskin's: their criterion for evolutionary stability did not rule out Tit-for-tat [4].

4. Proofs

In this section, we prove our results from the previous section.

4.1. **The time average distribution.**

Before we prove theorem 3.1, we need to understand the mechanisms behind the payoff $v_{s_1}(s_2)$. In this subsection, we prove a series of lemmas that characterize the time average distribution, and consequently $v_{s_1}(s_2)$.

Lemma 4.1. The time average distribution $\pi^{(a,b)}$, for any starting state (a, b), is a stationary distribution of the Markov chain. That is, if M is the transition matrix of the Markov chain, we have $M\pi^{(a,b)} = \pi^{(a,b)}$.

We state this lemma without proof, as it is a fairly standard result. Important to note is that the time average distribution is not necessarily a *unique* stationary distribution, as we make no assumptions that our Markov chain be ergodic.

Definition 4.2. A *strongly connected component* of a directed graph is a subgraph where there is a path from every node to every other node.

Definition 4.3. An *absorbing component* of a directed graph is a subgraph where there are no edges from vertices inside the component to vertices outside it.

We may also put both of the terms together and talk about absorbing strongly connected components, which, as shown by the next few lemmas, are useful.

Lemma 4.4. An absorbing strongly connected component has a unique time average distribution. That is, the time average distribution does not depend on the start state.

Proof. It is well known that a strongly connected (also known as irreducible) Markov chain has a unique stationary distribution. Since the time average distribution is stationary by lemma 4.1, it thus also has to be unique. □

Lemma 4.5. Let $\mathcal{A}$ be the set of absorbing strongly connected components of the s_1-s_2 Markov chain. For every component $S \in \mathcal{A}$, let $\pi^{(S)}$ be its unique time average distribution. Then,

$$\pi = \sum_{S\in\mathcal{A}} P(s_1\text{-}s_2 \text{ chain ends up in } S) \cdot \pi^{(S)}.$$

By $P(s_1\text{-}s_2 \text{ chain ends up in } S)$, we mean the probability of ending up in S given that we start in the initial states of the two strategies and perform a random walk in the Markov chain.

Proof. Condition all of the following on starting in the start state of s_1 and s_2. Let F be the fraction of time spent in state (c_1, c_2). If (c_1, c_2) is not in an absorbing strongly connected component, it is well known that the fraction of time in that state F will be 0. This is also what the lemma says. Suppose now that (c_1, c_2) is in an absorbing strongly connected component S. Let X be the event that the s_1-s_2 Markov chain ends up in S. By the law of total probability,

$$E[F] = E[F \mid X] \cdot P(X) + E[F \mid \text{not } X] \cdot P(\text{not } X)$$

If the random walk on the s_1-s_2 chain does not end up in the component S, but rather in another absorbing strongly connected component, the expected fraction of time in state (c_1, c_2) is clearly 0. Also, note that $E[F \mid X]$ is exactly $\pi^{(S)}_{c_1,c_2}$, and that $E[F]$ is exactly π_{c_1,c_2}. We thus find that

$$\pi_{c_1,c_2} = P(s_1\text{-}s_2 \text{ chain ends up in } S) \cdot \pi^{(S)}_{c_1,c_2}.$$

Note that $\pi^{(S')}_{c_1,c_2}$ is 0 for all other absorbing strongly connected components S'. With this, we have proved the lemma. □

Now that we know more about the time average distribution, we are also interested in proving that the payoff function behaves nicely.

Lemma 4.6. The limit

$$\lim_{p \to 0} v_{s_1}(s_2)$$

exists, for any strategies s_1 and s_2.

Proof. By definition,

$$v_{s_1}(s_2) = \pi \cdot r.$$

By lemma 4.1, π is a stationary distribution. In particular, if M is the transition matrix for the s_1-s_2 Markov chain, then π is an eigenvector of M with eigenvalue 1. This implies that π is in the nullspace of $M - I$. Thus, we can find π by solving for X in $(M - I)X = 0$. If we solve this using Gaussian elimination and back substitution, it is clear that the entries of π will be on the form $\frac{f(p)}{g(p)}$ where f and g are polynomials in p, since each entry of M is a second-degree polynomial in p. This is continuous for all p where $g(p) \neq 0$, and since g will have a finite degree it is therefore continuous in a small right neighborhood of 0. Finally, note that $v_{s_1}(s_2)$ is between S and T, which in conclusion means that the limit of it as p goes to 0 tends to a finite number, as desired. □

Lemma 4.7. For any strategy s,

$$v_s(s) \leq R$$

Proof. For notational simplicity, we will let s_1 and s_2 be two copies of strategy s. Then, $v_s(s) = v_{s_1}(s_2) = v_{s_2}(s_1)$. By definition, we have

$$v_{s_1}(s_2) = \sum \pi_{c_1,c_2} \cdot r(c_1, c_2)$$

and

$$v_{s_2}(s_1) = \sum \pi_{c_2,c_1} \cdot r(c_2, c_1).$$

Note that π_{c_1,c_2} and π_{c_2,c_1} refer to the same state, so we thus have

$$v_{s_1}(s_2) + v_{s_2}(s_1) = \sum \pi_{c_1,c_2} \cdot (r(c_1, c_2) + r(c_2, c_1))$$

which implies that

$$v_s(s) = \sum \left(\pi_{c_1,c_2} \cdot \frac{r(c_1, c_2) + r(c_2, c_1)}{2}\right).$$

Now, note that $r(c_1, c_2) + r(c_2, c_1) \in \{R + R, S + T, T + S, P + P\}$. Since $P < R$ and $T + S < 2R$, we thus find that

$$v_s(s) \leq \sum \pi_{c_1,c_2} \cdot R = R \sum \pi_{c_1,c_2} = R,$$

as desired. □

4.2. **Evolutionary Stability Requires Cooperation.** We are now finally ready to prove that to be evolutionarily stable, a strategy needs to cooperate with a clone of itself.

Proof of theorem 3.1. Suppose that the strategy s_1 is such that it is *not* true that

$$\lim_{p \to 0} v_{s_1}(s_1) = R.$$

By lemma 4.6 and lemma 4.7, this assumption implies that the limit is strictly less than R. Define $\gamma = \lim_{p \to 0} v_{s_1}(s_1)$. Then,

$$\gamma < R.$$

We want to prove that s_1 is not evolutionarily stable.

To do that, we want to prove that for all $p_0, \epsilon_0 \in (0, 1)$, there exists $p < p_0$ and $\epsilon < \epsilon_0$, such that s_1 is ϵ-invadable. We present a strategy s_2 that can invade s_1 for all p and ϵ that are sufficiently small.

The underlying idea is to construct s_2 such that s_1 can see no difference between itself and s_2, while s_2, on the other hand, can. If we succeed in doing so, we will have that $v_{s_1}(s_2) = v_{s_2}(s_1) = v_{s_1}(s_1)$, and can construct s_2 such that it always cooperates when it recognizes itself, thereby yielding $v_{s_2}(s_2) = R$. This would give s_2 a higher fitness than s_1. The rest of this proof executes this plan in detail.

We create the strategy s_2 as follows. First, we copy all of s_1 into s_2. Let $\mathcal{A}_1$ be the set of absorbing strongly connected components of s_1. The key idea, now, is to replace each original absorbing strongly connected component $H \in \mathcal{A}_1$ with a new absorbing component K_H, which has the capability of identifying itself.

We now construct a K_H for each H. Figure 3 shows the high-level construction of K_H. All of the steps succeed with probability $1 - O(p)$. The first step makes sure that K_H does not deviate from s_1 at all until s_1 has reached an absorbing strongly connected component. Once s_1 is in an absorbing strongly connected component, it cannot escape, and s_2 can also force it to end up in any of the states in the component, which means that even if s_1 after this point "figures out" that it is not playing against itself anymore, there's nothing it can do about it. The second step in the construction of K_H makes sure that if s_2 is playing against itself, it is in sync with its clone; that is, with probability $1 - O(p)$, both copies of s_2 will transition to the third component at the exact same time (although not necessarily in the same K_H). The third step now figures out what action s_1 would take at some large finite time T. Then, K_H outputs the exact opposite action at time T. At that point, if its opponent outputs s_1's expected action, it wants to transition into the absorbing strongly connected component H_{copy}, which is just an exact copy of H, but if its opponent outputs the opposite of that action, it has successfully recognized itself and enters an always cooperating state. Before transitioning into H_{copy}, s_2 makes sure that s_1 is in a state that is beneficial for s_2.

This concludes the high-level construction of s_2. We now want to prove that this s_2 can in fact invade s_1. To do that, we will first assume that the below four lemmas are correct, i.e., that the construction is possible and works as specified, and then finish the proof of our theorem using those assumptions. Once that's done, we will prove the four lemmas.

Lemma 4.8. We can construct a non-absorbing component with behavior indistinguishable from the behavior of component H, such that when we leave the component, s_1 will be in an absorbing strongly connected component with probability $1 - O(p)$.

Lemma 4.9. We can construct a non-absorbing component placed after the first component, such that if s_2 plays against itself, both clones transition from the second component to the third component of their respective K_H at the exact same time, with probability $1 - O(p)$.

Lemma 4.10. We can construct a non-absorbing component with a finite number of maximum steps that, supposing that s_2 plays against s_1, can figure out what action s_1 will take at the exact timestep following the transition out of the component. This component is the same for all H.

Lemma 4.11. We can construct a non-absorbing component placed after the third component that, supposing that s_2 plays against s_1, can transition into any given absorbing strongly connected component S in the s_1-s_2 chain that is a subset of $H_{\text{copy}} \times H'$ where H' is the absorbing strongly connected component that s_1 is in.

We now finish our proof, starting with a lemma concerning the payoffs.

Lemma 4.12. Given the above construction of s_2, the payoffs are as follows.

$$v_{s_1}(s_1) = \gamma \pm O(p)$$
$$v_{s_2}(s_2) = R \pm O(p),$$

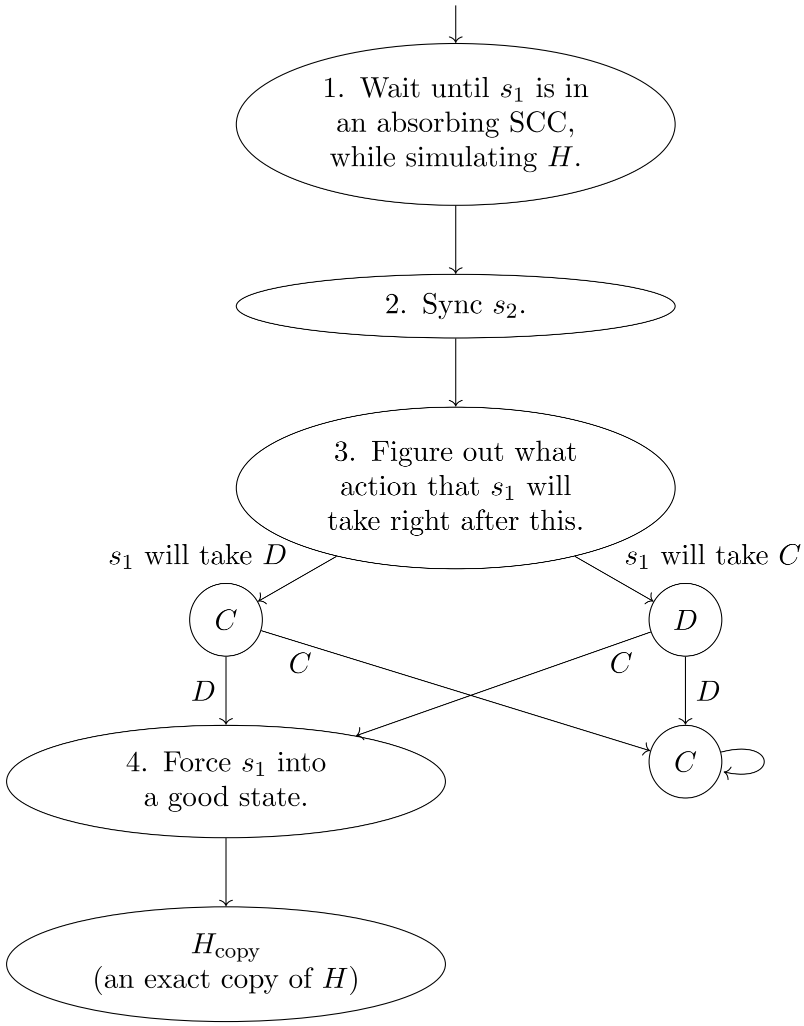

FIGURE 3. High level construction of K_H, the absorbing component replacing H when creating s_2 in the proof of theorem 3.1. Note that K_H contains exactly 2 absorbing strongly connected components. SCC means strongly connected component.

and either

$$\lim_{p\to 0} v_{s_2}(s_1) > \gamma,$$

or

$$v_{s_1}(s_2) = \gamma \pm O(p) \text{ and } v_{s_2}(s_1) = \gamma \pm O(p)$$

Proof. By our definition of γ, we have that $v_{s_1}(s_1) \leq \gamma \pm O(p)$.

Consider now the s_2-s_2 Markov chain. By lemma 4.9, both clones will transition into the third component at the exact same time. The third component spends only a finite amount of time by lemma 4.10, and is the same regardless of which K_H the strategies are in. Thus, after leaving the third component of K_H, if no mistakes have happened since leaving the second component, both clones of s_2 will be in exactly corresponding states, since they will always mirror each other. This happens with probability at least $(1-p)^N$ where N is the finite number of states in the third component, and thus with probability $1 - O(p)$. Thus, in the identifying stage, both clones will output the same action, and as we can see in fig. 3, they will then both transition to the always cooperating state with probability $1 - O(p)$. Therefore, $v_{s_2}(s_2) = R \pm O(p)$.

Consider now the case of s_1 playing against s_2. Suppose that s_2 ends up in the absorbing component K_H. By lemma 4.8, with probability $1 - O(p)$, s_2 is indistinguishable from s_1 until s_1 reaches an absorbing strongly connected component, which we call H'. This means that the probability of reaching the absorbing component $H' \times K_H$ in the s_1-s_2 Markov chain, is up to a factor of $1 - O(p)$ the same as if we had not replaced H by K_H in s_2, which is just the probability of reaching the absorbing component $H' \times H$ in the s_1-s_1 Markov chain. Thus,

$$P(s_1\text{-}s_2 \text{ ends up in } H' \times K_H) = (1 - O(p))P(s_1\text{-}s_1 \text{ ends up in } H' \times H)$$

Now, by lemma 4.10, the probability that s_2 enters the absorbing strongly connected component H_{copy} after having gotten to K_H is $1 - O(p)$ when playing against s_1. Thus,

$$(1)\quad P(s_1\text{-}s_2 \text{ ends up in } H' \times H_{\text{copy}}) = P(s_1\text{-}s_1 \text{ ends up in } H' \times H) \pm O(p).$$

By lemma 4.11, s_1 can choose to end up in any absorbing strongly connected component $S_{H',H}$ that is a subset of $H' \times H_{\text{copy}}$, with probability $1 - O(p)$. Naturally, s_1 chooses an $S^*_{H',H}$ as follows: among the S that have the maximum payoff for s_2, not caring about $O(p)$, we choose the one the minimizes the payoff for s_1. Let $v_s(S)$ denote the payoff that strategy s achieves in the absorbing strongly connected component S. Then,

(2)
$$v_{s_2}(S^*_{H',H}) \geq \sum_S P(s_1\text{-}s_1 \text{ ends up in } S \mid s_1\text{-}s_1 \text{ ends up in } H' \times H) \cdot v_{s_1}(S) \pm O(p)$$

since the right hand side is a weighted average of $v_{s_1}(S)$, which is the same as $v_{s_2}(S)$ in this case, which is what we maximized over when we chose $S^*_{H',H}$. Also, since s_1 could make sure to get into it with probability $1 - O(p)$, we have

$$(3)\quad P(s_1\text{-}s_2 \text{ ends up in } S^*_{H',H}) = P(s_1\text{-}s_2 \text{ ends up in } H' \times H_{\text{copy}}) \pm O(p).$$

Also, as a consequence, the $S^*_{H,H'}$ for all H and H' become the only absorbing strongly connected components with probabilities of ending up in them greater than $O(p)$.

Now, let $\mathcal{A}$ be the set of absorbing strongly connected components of the s_1-s_1 chain, and let $\mathcal{B}$ be the set of absorbing strongly connected components of the s_1-s_2 chain. We can now calculate $v_{s_2}(s_1)$.

$$\begin{aligned}
v_{s_2}(s_1) &= \sum_{S\in\mathcal{B}} P(s_1\text{-}s_2 \text{ ends up in } S)\cdot v_{s_2}(S) \\
&= \sum_{S^*_{H',H}} P(s_1\text{-}s_2 \text{ ends up in } S^*_{H',H})\cdot v_{s_2}(S^*_{H',H}) \pm O(p) && \textit{eq. } (3)\\
&= \sum_{S^*_{H',H}} P(s_1\text{-}s_2 \text{ ends up in } H'\times H_{\text{copy}})\cdot v_{s_2}(S^*_{H',H}) \pm O(p) && \textit{eq. } (1)\\
&= \sum_{S^*_{H',H}} P(s_1\text{-}s_1 \text{ ends up in } H'\times H)\cdot v_{s_2}(S^*_{H',H}) \pm O(p) && \textit{eq. } (2)\\
&\geq \sum_{S\in\mathcal{A}} P(s_1\text{-}s_1 \text{ ends up in } S)\cdot v_{s_1}(S) \pm O(p) \\
&= v_{s_1}(s_1) \pm O(p)
\end{aligned}$$

Thus, we have that $v_{s_2}(s_1) \geq \gamma \pm O(p)$. Suppose that $v_{s_2}(s_1) = \gamma \pm O(p)$. Then, we must have equality in eq. (2) for all H and H', which means that all S were equally good for s_2, in which case we chose $S^*_{H',H}$ to minimize the payoff for s_1. Following through with the exact same argument as above, but flipping the inequalities, we then find that $v_{s_1}(s_2) \leq \gamma \pm O(p)$.

■

Given lemma 4.12, we simply compute $F(s_2) - F(s_1)$, which we want to show is greater than 0. We have two cases. First, we assume that $\lim_{p\to 0} v_{s_2}(s_1) > \gamma$. Let $\lim_{p\to 0} v_{s_2}(s_1) = \beta$. Then,

$$\begin{aligned}
F(s_2) - F(s_1) &= \\
&= (1-\epsilon)\cdot v_{s_2}(s_1) + \epsilon\cdot v_{s_2}(s_2) - (1-\epsilon)\cdot v_{s_1}(s_1) - \epsilon\cdot v_{s_1}(s_2) \\
&= (1-\epsilon)(\beta \pm O(p)) + \epsilon(R \pm O(p)) - (1-\epsilon)(\gamma \pm O(p)) - \epsilon(v_{s_1}(s_2) \pm O(p)) \\
&\geq (1-\epsilon)(\beta-\gamma) + \epsilon(R - v_{s_1}(s_2)) \pm O(p)
\end{aligned}$$

Here, we can make ϵ small enough such that this becomes positive, since $\beta - \gamma > 0$. Now, in the second case, we assume that $v_{s_2}(s_1) = \gamma \pm O(p)$ and $v_{s_1}(s_2) \leq \gamma \pm O(p)$. Then,

$$\begin{aligned}
F(s_2) - F(s_1) &= \\
&= (1-\epsilon)\cdot v_{s_2}(s_1) + \epsilon\cdot v_{s_2}(s_2) - (1-\epsilon)\cdot v_{s_1}(s_1) - \epsilon\cdot v_{s_1}(s_2) \\
&\geq (1-\epsilon)(\gamma \pm O(p)) + \epsilon(R \pm O(p)) - (1-\epsilon)(\gamma \pm O(p)) - \epsilon(\gamma \pm O(p)) \\
&\geq \epsilon(R-\gamma) \pm O(p)
\end{aligned}$$

We know that $R - \gamma > 0$ by our initial assumption. Here, we can make p small enough such that this becomes positive.

In conclusion, our constructed s_2 has higher fitness than s_1. This proves that s_2 can invade s_1, and thus, that s_1 is not ϵ-invadable some value of $p < p_0$ and some $\epsilon < \epsilon_0$. In conclusion, then, s_1 is not evolutionarily stable, which concludes the proof of theorem 3.1.

□

We now return to the three lemmas that we left out, which detail the construction of s_2.

Proof of lemma 4.8. Let T_1 be the random variable designating the time at which s_1 transitions into an absorbing strongly connected component, counting from the time s_2 entered K_H. (In particular, if s_1 is already in an absorbing strongly connected component, $T_1 = 0$.) Let T_2 be the random variable designating the time at which s_2 transitions out of component 1 of K_H. To prove our lemma, we want to prove that with probability $1 - O(p)$, $T_1 \leq T_2$, and that during all the time that we are in component 1 of K_H, the output of s_2 is indistinguishable from H.

First, we deal with the indistinguishableness part. We create the component by copying H. Next, we define a new graph W, that we will design in the subsequent paragraphs, which will wait for s_1 to get into an absorbing component. This graph will have edges defined with the labels C and D still, but have no outputs at the states. The graph D will also have one designated "end state." We will then construct our component 1 for K_H by replacing every state $h \in H$ with a copy of D that we call D_h. There will be a C edge from node $u \in D_h$ to node $v \in D_{h'}$ if and only if there is a C edge from h to h', and from u to v in D. Similarly for the D edge. This is commonly known as the Kronecker or tensor product of the graphs D and H. The output at every state $u \in D_h$ is the same as the output of h. For the end state $e \in D_h$ for each h, we will add an edge with label both C and D, transitioning out of this component and into component 2.

It is clear that as long as we don't transition out of this component, the Kronecker product will ensure that the output will be indistinguishable from H. We now want to design D such that $T_1 \leq T_2$ with high probability.

Let d be the maximum length of a path from any state in s_1 to its closest absorbing strongly connected component. Then, for each state in s_1, there is a sequence of perceived inputs of length $\leq d$ (the *special sequence* of that state), such that s_1 upon seeing that sequence ends up in an absorbing strongly connected component. For all states, the probability that s_1 perceives its special sequence is $\geq p^d$, since, in the worst case, s_1 would need to make a mistake on exactly all inputs to perceive the special sequence. That means that at any single point in time, there is a probability $\geq p^d$ that s_1 will go into an absorbing strongly connected component within the subsequent d moves. Consider the geometric random variable $X_1 = d + \text{Geom}(p^d)$. For all t, then, we have that $P(X_1 = t) \leq P(T_1 = t)$. Thus, we may see X_1 as some sort of upper bound to T_1. 4 We now describe the construction of D. Consider a state $c_1 \in s_1$ and a state $h \in H$, and suppose that we let them evolve with no mistakes happening at all. Then, the outputs of c_1 will form an infinite string $w(c_1, h)$. Let there be $|s_1|$ states in s_1 and $|H|$ states in H. Now, create a string w^* of length $|s_1| \cdot |H|$, which differs in at least 1 place from each $w(c_1, h)$.[1] Thus, if s_1 ever outputs the string w^* when we are in this component, we know that at least one mistake has occurred, somewhere. Consequently, the probability that, at each timestep, w^* will be perceived by s_2 as the next sequence of outputs of s_1 is $\leq p$. If we repeat w^* $d+1$ times to form $(w^*)^{d+1}$, the probability that that string is perceived by s_2 as the next sequence of outputs of s_1 is $\leq p^{d+1}$. Now, create D as follows: it is a chain of states $d_1, d_2, \dots, d_{(d+1)|w^*|+1}$ where d_i transitions to d_{i+1} on seeing the ith character of the string $(w^*)^{d+1}$; d_i transitions to d_1 otherwise. $d_{(d+1)|w^*|+1}$ is the designated end state of D. Now, it is clear that each step, the probability that we will reach the end state in the next $(d+1)|w^*|+1$ steps is $\leq p^{d+1}$. Consider the geometric random variable $X_2 = (d+1)|w^*| + 1 + \mathrm{Geom}(p^{d+1})$. For all t, then, we have that $P(X_2 = t) \geq P(T_2 = t)$. Thus, we may see X_2 as a lower bound to T_2.

We now claim that the probability that $X_1 \leq X_2$ is $1-O(p)$, where X_1 and X_2 are independent. It suffices to show that $\mathrm{Geom}(p^d) \leq \mathrm{Geom}(p^{d+1})$ with high probability, and it further suffices to show that $\mathrm{Geom}(x) > \mathrm{Geom}(px)$ with probability $O(p)$, for some arbitrary x. This is easy to show. Consider the geometric variable with parameter px to consist of first an event of probability x happening, and then that counts as a real success with a probability p. Then, when comparing the two geometric variables, it is clear that both are equally likely to have an event of probability x happening first. Thus, with probability $\frac{1}{2}$, $\mathrm{Geom}(x)$ gets the first real success, with probability $\frac{1}{2} \cdot p$, $\mathrm{Geom}(px)$ gets the first real success, and with probability $\frac{1}{2}(1-p)$, we saw a fake success and will continue waiting for the next potential success. Clearly, then, $\mathrm{Geom}(x)$ will get the first success with probability $\frac{1}{2}/(\frac{1}{2} + \frac{1}{2}p)$. Thus, $\mathrm{Geom}(px)$ will get the first success with probability $p/(1+p) = O(p)$, which is exactly what we wanted to show.

This concludes the proof that component 1 is such that with probability $1 - O(p)$, s_1 will have reached an absorbing strongly connected component before s_2 continues to component 2 of K_H. □

Proof of lemma 4.9. This construction is relatively simple. It is a chain of states where each state transitions to the next on both input C and D (that is, deterministically). Let w_H^* be the string used in the proof of lemma 4.8 for component H. Let $W = \sum_H (d+1)|w_H^*|$. Then, the outputs of the first W states form the string w_H^* repeated $d+1$ times, for all w_H. The following $W+1$ states always output D. This is followed by one last state outputting C. The C state has a transition into itself on input D, and out of this component (to component 3 of K_H) on input C.

[1] This is the famous technique of diagonalization.

Suppose that s_2 plays against a clone s_2' of itself, and that s_2 enters an absorbing component K_H first. Then, by lemma 4.8, using the fact that s_2' is a copy of s_1 before it reaches an absorbing component, s_2 will not enter component 2 of K_H until s_2' enters an absorbing component $K_{H'}$, with probability $1 - O(p)$. When s_2 enters component 2, it will at some point output the entirety of the string $w^*_{H'}$. With probability $1-O(p)$, s_2' will then, by our construction of component 1, enter component 2. That is, s_2' enters the first part of component 2 before s_2 leaves the first part of component 2, which means that s_2 is at most W steps ahead of s_2. This means that s_2' will be in the defection part of the chain when s_2 reaches the cooperation state. Assuming no mistakes, then, s_2 will wait until s_2' also reaches the cooperation state, at which point they will both simultaneously leave the component. This happens with probability $1 - O(p)$, as desired. □

Proof of lemma 4.10. Let $\mathcal{A}_1$ be the set of all absorbing strongly connected components in s_1. Let L be the set of all tuples of (H, h) where $H \in \mathcal{A}_1$ and h is a possible state of H.

The third component is implemented as follows. When we talk about "at compile time" we refer to when we construct the finite automaton.

(1) Iterate over every pair of tuples (H, h) and (H', h') in L:
  (a) At compile time, we know what actions have been taken so far in this component, and thus know, if s_1 were in H in state h when s_2 started out this component, the state g that s_1 is in now, with high probability. Similarly supposing that s_1 were in H' in state h' when s_2 started out this component, we know the state g' where s_1 would be now.
  (b) At compile time, we also know whether there exists an input string w such that s_1 starting in state g perceiving w produces a different output sequence than if it started in state g', following only $1 - p$ transitions.
  (c) If no such w exists, continue the loop here.
  (d) If such a w exists, then output w deterministically here. Determine if the sequence of perceived actions corresponds to what would be expected from (H, h) or from (H', h').

In the end, there will be a subset U of L that corresponds to all tuples (H, h) that corresponded to the perceived sequence of actions in each comparison it was part of. By the way the loop is constructed, all (H, h) in U will produce the same high probability output on any input string; for our purposes, we can therefore just pick one canonical (H, h) from U.

Note that when we implement this procedure on a finite automaton, it becomes a large decision tree, which means that in the end, there will be one branch for each canonical (H, h), and we will be in one corresponding to where s_1 actually started off in the beginning of this component with probability $1 - O(p)$. In each such branch, we know, at compile time, which state s_1 is in at the end of the branch, and thus know what output that s_1

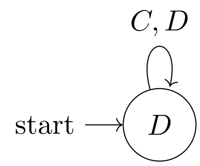

FIGURE 4. The always defecting All-D strategy.

will have in the next step, with probability $1 - O(p)$. This is exactly what we wanted, so we transition out of component 3. $\square$

Proof of lemma 4.11. Note that an absorbing strongly connected component S of the s_1-s_2 chain that is a subset of $H_{\text{copy}} \times H'$ can be gotten to by making s_1 transition to some state h_1 and s_2 transition to some state $h_2 \in H_{\text{copy}}$ where $(h_1, h_2) \in S$. This is simple. Note that s_1 is in a strongly connected component already, and that s_2 knows, by the previous component, which state s_1 is in. Thus, s_2 knows the path for s_1 to get to h_1, and can just output that. Then, s_1 will be in h_1, and s_2 can simply transition to h_2. This proves the lemma. $\square$

4.3. **Evolutionarily Stable Strategies Exist.** Unfortunately, we have no proof of conjecture 3.2.

We can note that the $2R > T + P$ condition is necessary. Otherwise, the All-D strategy, depicted in fig. 4, would be able to invade Pavlov. We see this by noting that $\lim_{p \to 0} v_{s_2}(s_1) = T + P$ if s_2 is All-D and s_1 is Pavlov, and that All-D is better against itself than Pavlov is against it.

We have some, admittedly weak, empirical evidence for this theorem to be true. First, Pavlov when played against itself gets payoff $R - O(p)$, and our proof of theorem 3.1 is very clearly relying heavily on letting p be very close to 0, so it seems unlikely that a similar proof would be able to prove that Pavlov is not evolutionarily stable. Moreover, Fudenberg and Maskin stated but did not prove a similar result in their version of the noisy regime [4]. We also wrote a Julia notebook for generating s_1-s_2 Markov chains and the time average distributions, and ran Pavlov against a few variants of Pavlov and other simple strategies like All-D, All-C and Tit-for-tat, and in all cases Pavlov performed better than our potential invaders; this can be found in the accompanying GitHub repository [5]. That repository also contains a document of unorganized notes that may provide hints for proving the conjecture.

5. Further Work

An obvious next step is to prove conjecture 3.2. We believe that it should not be too hard.

One may also consider different setups. For example, weakening definition 2.9 of evolutionary stability by saying that it is only required to not be ϵ-invadable for a fixed p, instead of for infinitesimally small p, would lead to a more interpretable model. Showing that theorem 3.1 still holds in this situation would be interesting.

One can also think of other kinds of noise: for example, a "failure of the mind," which could be modeled by a probability p of being transported to any random state in the automaton. This would automatically make the Markov chain strongly connected, which can potentially simplify a lot of the analysis.

References

[1] Axelrod, R. Effective choice in the prisoner's dilemma. *Journal of conflict resolution 24*, 1 (1980), 3–25.

[2] Axelrod, R., and Hamilton, W. D. The evolution of cooperation. *science 211*, 4489 (1981), 1390–1396.

[3] Binmore, K. G., and Samuelson, L. Evolutionary stability in repeated games played by finite automata. *Journal of economic theory 57*, 2 (1992), 278–305.

[4] Fundenberg, D., and Maskin, E. Evolution and cooperation in noisy repeated games. *The American Economic Review 80*, 2 (1990), 274–279.

[5] Lunnemark, A. Code for computing prisoner's dilemma markov chains. `https://github.com/arvid220u/prisonersdilemma`, 2020.

[6] Nowak, M. A., Sasaki, A., Taylor, C., and Fudenberg, D. Emergence of cooperation and evolutionary stability in finite populations. *Nature 428*, 6983 (2004), 646–650.