Forgiver Triumphs in Alternating Prisoner's Dilemma

The Harvard community has made this article openly available. Please share how this access benefits you. Your story matters.

Citation

Published Version
doi:10.1371/journal.pone.0080814

Citable link
http://nrs.harvard.edu/urn-3:HUL.InstRepos:11879285

Terms of Use
This article was downloaded from Harvard University’s DASH repository, and is made available under the terms and conditions applicable to Other Posted Material, as set forth at http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#LAA
Forgiver Triumphs in Alternating Prisoner’s Dilemma

Benjamin M. Zagorsky1,*, Johannes G. Reiter2,*, Krishnendu Chatterjee2, Martin A. Nowak1,3*

1 Program for Evolutionary Dynamics, Harvard University, Cambridge, Massachusetts, United States of America, 2 IST Austria (Institute of Science and Technology Austria), Klosterneuburg, Lower Austria, Austria, 3 Department of Mathematics, Department of Organismic and Evolutionary Biology, Harvard University, Cambridge, Massachusetts, United States of America

Abstract

Cooperative behavior, where one individual incurs a cost to help another, is a widespread phenomenon. Here we study direct reciprocity in the context of the alternating Prisoner’s Dilemma. We consider all strategies that can be implemented by one and two-state automata. We calculate the payoff matrix of all pairwise encounters in the presence of noise. We explore deterministic selection dynamics with and without mutation. Using different error rates and payoff values, we observe convergence to a small number of distinct equilibria. Two of them are uncooperative strict Nash equilibria representing always-defect (ALLD) and Grim. The third equilibrium is mixed and represents a cooperative alliance of several strategies, dominated by a strategy which we call Forgiver. Forgiver cooperates whenever the opponent has cooperated; it defects once when the opponent has defected, but subsequently Forgiver attempts to re-establish cooperation even if the opponent has defected again. Forgiver is not an evolutionarily stable strategy, but the alliance, which it rules, is asymptotically stable. For a wide range of parameter values the most commonly observed outcome is convergence to the mixed equilibrium, dominated by Forgiver. Our results show that although forgiving might incur a short-term loss it can lead to a long-term gain. Forgiveness facilitates stable cooperation in the presence of exploitation and noise.

Introduction

A cooperative dilemma arises when two cooperators receive a higher payoff than two defectors and yet there is an incentive to defect [1,2]. The Prisoner’s Dilemma [3–9] is the strongest form of a cooperative dilemma, where cooperation requires a mechanism for its evolution [10]. A mechanism is an interaction structure that specifies how individuals interact to receive payoff and how they compete for reproduction. Direct reciprocity is a mechanism for the evolution of cooperation. Direct reciprocity means there are repeated encounters between the same two individuals [11–37]. The decision whether or not to cooperate depends on previous interactions between the two individuals. Thus, a strategy for the repeated Prisoner’s Dilemma (or other repeated games) is a mapping from any history of the game into what to do next. The standard theory assumes that both players decide simultaneously what do for the next round. But another possibility is that the players take turns when making their moves [38–40]. This implementation can lead to a strictly alternating game, where the players always choose their moves in turns, or to a stochastically alternating game, where in each round the player to move is chosen at random next is selected probabilistically. Here we investigate the strictly alternating game.

We consider the following scenario. In each round a player can pay a cost, c, for the other player to receive a benefit, b, where \( b > c > 0 \). If both players cooperate in two consecutive moves, each one gets a payoff, \( b - c \), which is greater than the zero payoff they would receive for mutual defection. But if one player defects, while the other cooperates, then the defector gets payoff, \( b \), while the cooperator gets the lowest payoff, \( -c \). Therefore, over two consecutive moves the payoff structure is the same as in a Prisoner’s Dilemma: \( b > b - c > 0 > -c \). Thus, this game is called “alternating Prisoner’s Dilemma” [29,39].

We study the strictly alternating Prisoner’s Dilemma in the presence of noise. In each round, a player makes a mistake with probability \( \epsilon \) leading to the opposite move. We consider all strategies that can be implemented by deterministic finite state automata [41] with one or two states. These automata define how a player behaves in response to the last move of the other player. Thus we consider a limited strategy set with short-term memory. Finite-state automata have been used extensively to study repeated games [42–45] including the simultaneous Prisoner’s Dilemma. In our case, each state of the automaton is labeled by \( C \) or \( D \). In state \( C \) the player will cooperate in the next move; in state \( D \) the player will defect. Each strategy starts in one of those two states. Each state has two outgoing transitions (either to the same or to the other state); one transition specifies what happens if the opponent has cooperated (labeled with \( c \)) and one if the opponent has defected (labeled with \( d \)). There are 26 automata encoding unique strategies (Fig. 1). These strategies include ALLC, ALLD, Grim, tit-for-tat (TFT), and win-stay lose-shift (WSLS).

ALLC (\( S_{26} \)) and ALLD (\( S_{1} \)) are unconditional strategies (see Fig. 1 and Supporting File S1 for strategy names and their indexing). ALLC always cooperates while ALLD always defects.


Editor: Attila Csikász-Nagy, Fondazione Edmund Mach, Research and Innovation Centre, Italy

Received: August 23, 2013; Accepted: October 6, 2013; Published: December 12, 2013

Copyright: © 2013 Zagorsky et al. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Funding: This work was supported by grants from the John Templeton Foundation, ERC Start grant (279307: Graph Games), FWF NFN Grant (No S11407N23 RISE), FWF Grant (No P23499N23), and a Microsoft faculty fellows award. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: martin_nowak@harvard.edu

These authors contributed equally to this work.
Figure 1. Deterministic strategies in the Prisoner’s Dilemma. Each automaton defines a different strategy for how a player behaves during the game. If a player is in state C, she will cooperate in the next move; if she is in state D, then she will defect. The outgoing transitions of a state define how the state of an automaton will change in response to cooperating (label c) or defecting (label d) of the opponent. The left state with the
Both strategies are implemented by a one-state automaton (Fig. 1). The strategy Grim starts and stays in state C as long as the opponent cooperates. If the opponent defects, Grim permanently moves to state D with no possibility to return. TFT (S15) starts in state C and subsequently does whatever does the opponent did in the last round [5]. This simple strategy is very successful in an error-free environment as it promotes cooperative behavior but also avoids exploitation by defectors. However, in a noisy environment TFT achieves a very low payoff against itself since it can only recover from a single error by another error [46]. WSLS (S16) has the ability to correct errors in the simultaneous Prisoner’s Dilemma [47]. This strategy also starts in state C and moves to state D whenever the opponent defects. From state D strategy WSLS switches back to cooperation only if another defection occurs. In other words, WSLS stays in the current state whenever it has received a high payoff, but moves to the other state, if it has received a low payoff.

We can divide these 26 strategies into four categories: (i) sink-state C (ssC) strategies, (ii) sink-state D (ssD) strategies, (iii) suspicious dynamic strategies, and (iv) hopeful dynamic strategies. Sink-state strategies always-cooperate or always-defect either from the beginning or after some condition is met. They include ALLC, ALLD, Grim and variations of them. There are eight sink-state strategies in total. Suspicious dynamic strategies start with defection and then move between their defective and cooperative state depending on the other player’s decision. Hopeful dynamic strategies do the same, but start with cooperation. There are nine strategies in each of these two categories. For each suspicious dynamic strategy there is a hopeful counterpart. Some of the dynamic strategies do little to optimize their score. For example, Alternator (S22) switches between cooperation and defection on each move. But a subset of dynamic strategies are of particular interest: Forgiver (S14), TFT, WSLS, and their suspicious counterparts (S4, S8, and S12). These strategies have the design element to stay in state C if the opponent has cooperated in the last round but move to state D if the opponent has defected; we call this element the conditional cooperation element (Fig. S1 in File S1) which ensures the benefit of mutual cooperation but also avoids being exploited by defection-heavy strategies.

We can divide these 26 strategies into four categories: (i) sink-state C (ssC) strategies, (ii) sink-state D (ssD) strategies, (iii) suspicious dynamic strategies, and (iv) hopeful dynamic strategies. The automata with the blue background shading contain a conditional cooperation element (Fig. S1 in File S1) which ensures the benefit of mutual cooperation but also avoids being exploited by defection-heavy strategies.

doi:10.1371/journal.pone.0080814.g001

Figure 2. Performance of the conditional cooperators in the presence of noise. An asterisk after a move indicates that this move was caused by an error. When the conditional cooperators are playing against a copy of themselves, Forgiver performs very well as it can recover from an accidental defection within three rounds. Against defection-heavy strategies like Grim and ALLD, Forgiver gets exploited in each second round. Both TFT and WSLS are not error correcting as they are unable to recover back to cooperation after an unintentional mistake. Only another mistake can enable them to return to cooperative behavior. When Grim plays against itself and a single defection occurs, it moves to the defection state with no possibility of returning to cooperation.

doi:10.1371/journal.pone.0080814.g002
neither strategy is error correcting (Fig. 2). Interestingly TFT vs WLSL yields good payoff for both strategies, because their interaction is error correcting.

In the following, we study evolutionary game dynamics [48–50] with the replicator equation. The frequency of strategy $S_i$ is denoted by $x_i$. At any one time we have $\sum_{i=1}^{N} x_i = 1$, where $n = 26$ is the number of strategies. The frequency $x_i$ changes according to the relative payoff of strategy $S_i$. We evaluate evolutionary trajectories for many different initial frequencies. The trajectories start from $10^4$ uniformly distributed random points in the 26-simplex.

Typically, we do not find convergence to one of the strict Nash equilibria (Fig. 3 b). In only 5% of the cases the trajectories converge to the pure ALLD equilibrium and in 18% of the cases the trajectories converge to the pure Grim equilibrium. However, in 77% of the cases we observe convergence to a mixed equilibrium of several strategies, dominated by Forgiver with a population share of 77% of the cases we observe convergence to a mixed equilibrium of Forgiver and Grim ($S_{17}$), both denoted in bold.

doi:10.1371/journal.pone.0080814.t001

Table 1. Payoff matrix for the most relevant strategies.

<table>
<thead>
<tr>
<th></th>
<th>ALLD</th>
<th>Forgiver</th>
<th>TFT</th>
<th>WLSL</th>
<th>Grim</th>
<th>ALLC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLD</td>
<td>10.0</td>
<td>148.4</td>
<td>24.8</td>
<td>144.9</td>
<td>11.5</td>
<td>280.0</td>
</tr>
<tr>
<td>Forgiver</td>
<td>-36.1</td>
<td>174.8</td>
<td>163.5</td>
<td>166.9</td>
<td>-12.7</td>
<td>194.3</td>
</tr>
<tr>
<td>TFT</td>
<td>5.1</td>
<td>178.1</td>
<td>104.5</td>
<td>176.7</td>
<td>24.5</td>
<td>194.5</td>
</tr>
<tr>
<td>WLSL</td>
<td>-35.0</td>
<td>169.1</td>
<td>162.3</td>
<td>106.5</td>
<td>-12.0</td>
<td>230.9</td>
</tr>
<tr>
<td>Grim</td>
<td>9.5</td>
<td>152.0</td>
<td>40.5</td>
<td>148.8</td>
<td>28.1</td>
<td>262.9</td>
</tr>
<tr>
<td>ALLC</td>
<td>-80.0</td>
<td>177.2</td>
<td>176.6</td>
<td>67.3</td>
<td>-28.8</td>
<td>190.0</td>
</tr>
</tbody>
</table>

Excerpt from the payoff matrix with the most relevant strategies when the benefit value $b = 3$ ($c = 1$), the error rate $\epsilon = 5\%$, and the number of rounds in each game $L = 100$. There are two pure Nash equilibria in the full payoff matrix: ALLD ($S_1$) and Grim ($S_{17}$), both denoted in bold.

For the calculations see File S1: Section 2 and Fig. S8–S10. From these results we obtain that ALLD (or ssD strategies) cannot invade Forgiver if

$$b \geq \frac{2 + \epsilon - \epsilon^2}{1 - 2\epsilon}.$$ 

This result holds for any error rate, $\epsilon$, between 0 and 1/2 (Fig. 4 d).

Discussion

Our results imply an indisputable strength of the strategy Forgiver in the alternating Prisoner’s Dilemma in the presence of noise. For a wide range of parameter values, Forgiver is the dominating strategy of the cooperative equilibrium, having a population share of more than half in all investigated scenarios. Essential for the success of a cooperative strategy in the presence of noise is how fast it can recover back to cooperation after a mistake, but at the same time, also avoid excessive exploitation by defectors. The conditional cooperation element is crucial for the triumph of Forgiver. Even though, also TFT and WLSL contain this element, which allows them to cooperate against cooperative strategies without getting excessively exploited by defectors, these strategies are not as successful as Forgiver, because of their inability to correct errors. Grim also possesses this conditional cooperation element. However, noise on the part of Grim’s opponent will inevitably cause Grim to switch to always-defect. It is Grim’s ability to conditionally cooperate for the first handful of turns that provides a competitive advantage over pure ALLD such that the strict Nash equilibrium ALLD can only rarely arise.

The other strategies appearing in the Forgiver equilibrium for the cases of $b = 3$, $b = 4$, and $b = 5$ are Paradoxical Grateful ($S_5$), Grateful ($S_6$), Suspicious ALLC ($S_{13}$), and ALLC ($S_{16}$). All of them are ssC strategies that, in the presence of noise, behave like ALLC

against Grim can become higher than the payoff of Grim against itself. In other words, Grim can be invaded by ALLC. Hence, instead of the pure Grim equilibrium we observe a mixed equilibrium between Grim and ALLC (Fig. S4b–d in File S1).

We check the robustness of the observed equilibria by incorporating mutation to the replicator equation. We find that both the ALLD and the rare Suspicious Forgiver equilibrium are unstable. In the presence of mutation the evolutionary trajectories lead away from ALLD to Grim and from Suspicious Forgiver to Forgiver (see Fig. S6 and S7 in File S1). The Grim equilibrium and the Forgiver equilibrium remain stable. We note that this asymptotic stability is also due to the restricted strategy space. In [51] it has been shown that in the simultaneous Prisoner’s Dilemma with an unrestricted strategy space, no strategy is robust against indirect invasions and hence, no evolutionarily stable strategy can exist.

Essential for the stability in our model is that Forgiver can resist invasion by ssD strategies ($S_1, S_7, S_{14}, S_{26}$), because Forgiver does better against itself than the ssD strategies do against Forgiver (Table 1). However, Forgiver can be invaded by ssC strategies and TFT. But, since TFT performs poorly against itself and ssC strategies are exploited by WLSL (Table 1), all these strategies can coexist in the Forgiver equilibrium. Stable alliances of cooperative strategies have also been found in the context of the Public Goods Game [52] and indirect reciprocity [53]. More detailed results and equilibrium analysis for a wide range of parameter values for $\epsilon$ and $L$ are provided in File S1 (Tables S1–S14 and Figures S2–S7).
after the first few moves. The strategy ALLC does very well in combination with Forgiver. Nevertheless, ALLC itself appears rarely. Perhaps because of Paradoxic Grateful, which defects against ALLC for many moves in the beginning, whereas Suspicious ALLC puts Paradoxic Grateful into its cooperating state immediately. One might ask why these ssC strategies do not occupy a larger population share in the cooperative equilibrium. The reason is the presence of exploitative strategies like WSLS which itself is a weak strategy in this domain. If only Forgiver was present, WSLS would be quickly driven to extinction; WSLS does worse against itself and Forgiver than Forgiver does against WSLS and itself (see Table 1). But WSLS remains in the Forgiver equilibrium because it exploits the ssC strategies. Interestingly, higher error rates can decrease the probability to converge to the cooperative equilibrium dramatically and hence prevent the evolution of any cooperative behavior (Fig. 4c).

Grim and Forgiver are similar strategies, the difference being, in the face of a defection, Forgiver quickly returns to cooperation whereas Grim never returns. An interesting interpretation of the relationship is that Grim never forgives while Forgiver always does. Thus, the clash between Grim and Forgiver is actually a test of the viability of forgiveness under various conditions. On the one hand, the presence of noise makes forgiveness powerful and essential. On the other hand, if cooperation is not valuable enough, forgiveness can be exploited. Moreover, even when cooperation is valuable, but the population is ruled by exploiters, forgiveness is not a successful strategy. Given the right conditions, forgiveness makes cooperation possible in the face of both exploitation and noise.

These results demonstrate a game-theoretic foundation for forgiveness as a means of promoting cooperation. If cooperation is valuable enough, it can be worth forgiving others for past wrongs in order to gain future benefit. Forgiving incurs a short-term loss but ensures a greater long-term gain. Given all the (intentional or unintentional) misbehavior in the real world, forgiveness is essential for maintaining healthy, cooperative relationships.

Methods

Strategy space

We consider deterministic finite automata [41] (DFA) with one and two states. There are two one-state automata which encode...
the strategies always-defect (ALLD) and always-cooperate (ALLC). In total, there are 32 two-state automata encoding strategies in our game: two possible arrangements of states (CD, DC) and 16 possible arrangements of transitions per arrangement of states. For 8 of these 32 automata, the second state is not reachable, making them indistinguishable from a one-state automata. Since we already added the one-state automata to our strategy space, these 8 can be ignored. The remaining 24 two-state automata encode

Figure 4. Robustness of results across various benefit values and error rates. a | Convergence probability to the Forgiver equilibrium of a uniform-random point in the 26-simplex. Note that for higher error rates (increasing noise-level), the probability to converge to the cooperative equilibrium is much lower. b | Population share of Forgiver ($S_{14}$) in the Forgiver equilibrium. Observe the relationship between the higher error rates and the lower population share of Forgiver. c | Population share of sink-state C strategies ($S_1, S_5, S_9, S_{13}, S_{26}$) in the Forgiver equilibrium. Higher error rates lead to higher proportions of unconditional cooperators. d | In the infinitely repeated game, for all value pairs of $b/c$ and $\epsilon$ in the blue shaded area, ALLD cannot invade Forgiver since the average payoff of Forgiver playing against itself is higher than the average payoff of ALLD against Forgiver (see Inequality (1)).

doi:10.1371/journal.pone.0080814.g004

Table 2. Equilibrium frequencies.

<table>
<thead>
<tr>
<th></th>
<th>ALLD</th>
<th>Grim</th>
<th>Forgiver</th>
<th>SForgiver</th>
</tr>
</thead>
<tbody>
<tr>
<td>$b=2$</td>
<td>15%</td>
<td>52%</td>
<td>33%</td>
<td>&lt;1%</td>
</tr>
<tr>
<td>$b=3$</td>
<td>5%</td>
<td>18%</td>
<td>77%</td>
<td>0%</td>
</tr>
<tr>
<td>$b=4$</td>
<td>2%</td>
<td>7%</td>
<td>90%</td>
<td>1%</td>
</tr>
<tr>
<td>$b=5$</td>
<td>1%</td>
<td>3%</td>
<td>93%</td>
<td>3%</td>
</tr>
</tbody>
</table>

Proportions in which the four equilibria were reached from $10^6$ uniformly distributed random points in the 26-simplex. In the case of $b=2$, the mixed Forgiver equilibrium has a different composition than in the cases of $b = 3, b = 4, b = 5$. Parameter values: costs $c=1$, error rate $\epsilon = 0.05$, number of rounds per game $L=100$.

doi:10.1371/journal.pone.0080814.t002

Table 3. Analytical results in the infinitely alternating Prisoner’s Dilemma.

<table>
<thead>
<tr>
<th></th>
<th>ALLD</th>
<th>Forgiver</th>
<th>ALLC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLD</td>
<td>$c(b-c)$</td>
<td>$\frac{1-c^2}{2} - \epsilon c$</td>
<td>$b-c(b+c)$</td>
</tr>
<tr>
<td>Forgiver</td>
<td>$c b - \frac{1-c^2}{2} - \epsilon c$</td>
<td>((b-c)\frac{1+c(1-c)}{1+3c-c^2})</td>
<td>$b(1-c) - c(1-c)\frac{1-c(1-c)}{1+\epsilon}$</td>
</tr>
<tr>
<td>ALLC</td>
<td>$c b - (1-c)$</td>
<td>$\frac{1-c+\epsilon^2}{1+\epsilon}$</td>
<td>$b(1-c)(1+c)$</td>
</tr>
</tbody>
</table>

Analytical results of the average payoff per round in the infinitely alternating Prisoner’s Dilemma for ALLD ($S_1$), Forgiver ($S_{14}$), and ALLC ($S_{26}$) playing against each other. Derivations are provided in File S1 (section 2).

doi:10.1371/journal.pone.0080814.t003
distinct strategies in our game. Hence, in total we have 26
deterministic strategies in the alternating Prisoner’s Dilemma
(Fig. 1).

Generation of the payoff matrix

In each round of the game a player can either cooperate or
defect. Cooperation means paying a cost, $c$, for the other player to
receive a benefit, $b$. Defection means paying no cost and
distributing no benefit. If $b > c > 0$ and we sum over two
consecutive moves (equivalent to one round), the game is a
Prisoner’s Dilemma since the following inequality is satisfied:
$b > b - c > 0 > -c$. In other words, in a single round it is best to
defect, but cooperation might be fruitful when playing over
multiple rounds. Furthermore also $2(b - c) > b - c$ holds, and
hence mutual cooperation results in a higher payoff than
alternating between cooperation and defection. The second
inequality ensures that sustained cooperation results in a higher
payoff than alternation between cooperation and defection.

For each set of parameters (number of rounds $L$, error rate $\epsilon$,
benefit value $b$, and costs $c$), we generate a $26 \times 26$ payoff matrix $A$
where each of the 26 distinct strategies is paired with each other.
The entry $a_{ij}$ in the payoff matrix $A$ gives the payoff of strategy $S_i$
playing against strategy $S_j$. Based on the average of which strategy
(player) goes first, we define the initial state distribution of both
players as a row vector $Q_{S_i \times S_j}$. Since the players do not observe
when they have made a mistake (i.e., the faulty player does not
move to the corresponding state of the erroneous action which he
has accidentally played), the state space consists of sixteen states
namely $CC, CD, DC, DD, D' C, D' D, C' C, C' D, C' D', \ldots$,
$C' C'$. The star after a state indicates that the player accidentally
played the opposite move as intended by her current state.

Each game consists of $L$ moves of both player. In each move, a
player makes a mistake with probability $\epsilon$ and thus implements the
opposite move of what is specified by her strategy (automaton). We
denote $1 - \epsilon$ by $\tilde{\epsilon}$. Although, the players do not observe their
mistakes, the payoffs depend on the actual moves. This setting
relates to imperceptible implementation errors [16,18,29,45] [see
section 3 in File S1 for a discussion on error types]. The payoffs
corresponding to their moves in the different states are given by
the column vector $U$.

Next, we define a $16 \times 16$ transition matrix $M_{S_i \times S_j}$ for each pair
of strategies $S_i$, $S_j$. The entries of the transition matrix are given
by the probabilities to move from eachone state of the sixteen
states (defined above) to the next:

$$
M_{S_i \times S_j} = 
\begin{pmatrix}
p_1 p_2 1 & p_1 (1-p_2) 1 & (1-p_1) p_3 1 & (1-p_1) (1-p_3) 1 & p_1 p_3 2 & p_1 (1-p_3) 2 & (1-p_1) p_4 2 & (1-p_1) (1-p_4) 2 & \cdots & \cdots & \cdots & \cdots
\end{pmatrix}
$$

where the quadruple $[39] (p_1, p_2, p_3, p_4)$ defines the probabilities of
strategy $S_j$ to cooperate in the observed states $CC, CD, DC,$ and
$DD$ (errors remain undetected by the players). Respectively, the
quadruple $(p_1', p_2', p_3', p_4')$ encodes the strategy $S_j$. For example,$(1-p_1) p_3 2$ is the probability to move from state $DD$ to state $C' C$.
A deterministic strategy is represented as a quadruple where each
$p_i \in (0,1)$.

Using the initial state distribution $Q_{S_i \times S_j}$, the transition matrix
$M_{S_i \times S_j}$, and the payoff vector $U$, we calculate the payoff $a_{ij}$ of
strategy $S_i$ playing against strategy $S_j$ via a Markov Chain:

$$
a_{ij} = Q_{S_i \times S_j} \times \sum_{k=0}^{L-1} M_{S_i \times S_j}^k \times U. \quad (3)
$$

Applying equation (3) to each pair of strategies, we obtain the
entire payoff matrix $A$ for a given set of parameter values.
Although we use deterministic strategies, the presence of noise
implies that the game that unfolds between any two strategies is
described by a stochastic process. Payoff matrices for benefit values
of $b = 2, b = 3, b = 4$, and $b = 5$, for error rates of $\epsilon = 0.01, \epsilon = 0.05,$
and $\epsilon = 0.1$, and for game length of $L = 10, L = 100$, and $L = 1000$
are provided in File S1 (Tables S1–S8).

Evolution of strategies. The strategy space spans a 26-simplex which we explore via the replicator equation [48–50] with
and without mutations. The frequency of strategy $S_i$ is given by $x_i$. At any time $\sum_{i=1}^{n} x_i = 1$ holds where $n = 26$ is the number of
strategies. The average payoff (fitness) for strategy $S_i$ is given by

$$
f_i = \sum_{j=1}^{n} a_{ij} x_j. \quad (4)
$$

The frequency of strategy $S_i$ changes according to the differential
equation

$$
x_i = x_i (f_i - \bar{u}) + u \left( \frac{1}{n} - x_i \right) \quad (5)
$$

where the average population payoff is $\bar{u} = \sum_{i=1}^{n} f_i x_i$ and $u$ is the
mutation rate. Mutations to each strategy are equally likely; for
non-uniform mutation structures see [54]. Using the differential
equation (5), defined on the $n$-simplex [here, $n = 26$], we study the
evolutionary dynamics in the alternating Prisoner’s Dilemma for
different initial conditions (i.e., random initial frequencies of
the strategies). We generate a uniform-random point in the
$n$-simplex by taking the negative logarithm of $n$ random numbers in
$(0,1)$, then normalizing these numbers such that they sum to 1,
and using the normalized values as the initial frequencies of the $n$
strategies [53].

Computer simulations. Our computer simulations are
implemented in Python and split into three programs. The first
program generates the $26 \times 26$ payoff matrix for each set of
parameters. The second program simulates the deterministic
selection dynamics starting from uniform-random points in the 26-
simplex. The third program performs statistical analysis on the
results of the second program. The code is available at http://pub.
ist.ac.at/~jeiter upon request [56].

Supporting Information

File S1 Detailed description of the model and the
strategies; Simulation results and equilibrium analysis for a wide range of parameter values; Calculations for
the infinitely-repeated game; Implementation of errors;
Includes Tables S1–S14 and Figures S1–S10.

(PDF)

Acknowledgments

We thank Sasha Rubin and Sebastian Novak for helpful comments and
suggestions.
Author Contributions
Conceived and designed the experiments: BMZ JGR MAN. Performed the experiments: BMZ JGR MAN. Analyzed the data: BMZ JGR MAN. Wrote the paper: BMZ JGR MAN. Provided input to the manuscript: KC.

References