# Evolution Reinforces Cooperation with the Emergence of Self-Recognition Mechanisms: an empirical study of the Moran process for the iterated Prisoner's dilemma using reinforcement learning

Vincent Knight    Marc Harper    Nikoleta E. Glynatsi    Owen Campbell

**Abstract**

We present insights and empirical results from an extensive numerical study of the evolutionary dynamics of the iterated prisoner's dilemma. Fixation probabilities for Moran processes are obtained for all pairs of 164 different strategies including classics such as TitForTat, zero determinant strategies, and many more sophisticated strategies. Players with long memories and sophisticated behaviours outperform many strategies that perform well in a two player setting. Moreover we introduce several strategies trained with evolutionary algorithms to excel at the Moran process. These strategies are excellent invaders and resistors of invasion and in some cases naturally evolve handshaking mechanisms to resist invasion. The best invaders were those trained to maximize total payoff while the best resistors invoke handshake mechanisms. This suggests that while maximizing individual payoff can lead to the evolution of cooperation through invasion, the relatively weak invasion resistance of payoff maximizing strategies are not as evolutionarily stable as strategies employing handshake mechanisms.

## 1 Introduction

The Prisoner's Dilemma (PD) [20] is a fundamental two player game used to model a variety of strategic interactions. Each player chooses simultaneously and independently between cooperation (C) or defection (D). The payoffs of the game are defined by the matrix $\begin{pmatrix} R & S \\ T & P \end{pmatrix}$, where $T > R > P > S$ and $2R > T + S$. The PD is a one round game, but is commonly studied in a manner where the prior outcomes matter. This repeated form is called the Iterated Prisoner's Dilemma (IPD). As described in [10, 29, 44] a number of strategies have been developed to take advantage of the history of play. Recently, some strategies referred to as zero determinant (ZD) strategies [44] can manipulate some players through extortionate mechanisms.

The Moran Process [40] is a model of evolutionary population dynamics that has been used to gain insights about the evolutionary stability in a number of settings (more details given in Section 1.1). Several earlier works have studied iterated games in the context of the prisoner's dilemma [43, 50], however these often make simplifying assumptions or are limited to classes of strategies such as memory-one strategies that only use the previous round of play.

This manuscript provides a detailed numerical analysis of agent-based simulations of **164** complex and adaptive strategies for the IPD. This is made possible by the Axelrod library [51], an effort to provide software for reproducible research for the IPD. The library now contains over 186 parameterized strategies including classics like TitForTat and WinStayLoseShift, as well as recent variants such as OmegaTFT, zero determinant and other memory one strategies, strategies based on finite state machines, lookup tables, neural networks, and other machine learning based strategies, and a collection of novel strategies. Not all strategies have been considered for this study: excluded are those that make use of knowledge of the number of turns in a match and others that have a high computational run time. The large number of strategies are available thanks to the open source nature of the project with over 50 contributors from around the world, made by programmers and researchers [29]. Three of the considered strategies are finite state machines trained specifically for Moran processes (described further in Section 1.2).

In addition to providing a large collection of strategies, the Axelrod library can conduct matches, tournaments and population dynamics with variations including noise and spatial structure. The strategies and simulation frameworks are automatically tested to an extraordinarily high degree of coverage in accordance with best research software practices.

Using the Axelrod library and the many strategies it contains, we obtain fixation probabilities for all pairs of strategies, identifying those that are effective invaders and those resistant to invasion, for population sizes $N = 2$ to $N = 14$. Moreover we present a number of strategies that were created via reinforcement algorithms (evolutionary and particle swarm algorithms) that are among the best invaders and resistors of invasion known to date, and show that handshaking mechanisms naturally arise from these processes as an invasion-resistance mechanism.

Recent work has argued that agent-based simulations can provide insights in evolutionary game theory not available via direct mathematical analysis [2]. The results and insights contained in this paper would be difficult to derive analytically.

In particular the following questions are addressed:

1. What strategies are good invaders?
2. What strategies are good at resisting invasion?
3. How does the population size affect these findings?

While the results agree with some of the published literature, it is found that:

1. Zero determinant strategies are not particularly effective for $N > 2$
2. Complex strategies can be effective, and in fact can naturally evolve through evolutionary processes to outperform designed strategies.
3. The strongest resistors specifically evolve or have a handshake mechanism.
4. Strong invaders are generally cooperative strategies that do not defect first but retaliate to varying degrees of intensity against strategies that defect.
5. Strategies evolved to maximize their total payoff can be strong invaders and achieve mutual cooperation with many other strategies.

## 1.1 The Moran Process

Figure 1 shows a diagrammatic representation of the Moran process, a stochastic birth death process on a finite population in which the population size stays constant over time. Individuals are **selected** according to a given fitness landscape. Once selected, the individual is reproduced and similarly another individual is chosen to be removed from the population. In some settings mutation is also considered but without mutation (the case considered in this work) this process will arrive at an absorbing state where the population is entirely made up of players of one strategy. The probability with which a given strategy takes over a population is called the *fixation probability*. A more detailed analytic description of this is given in Section 2. In our simulations offspring do not inherit any knowledge or history from parent replicants.

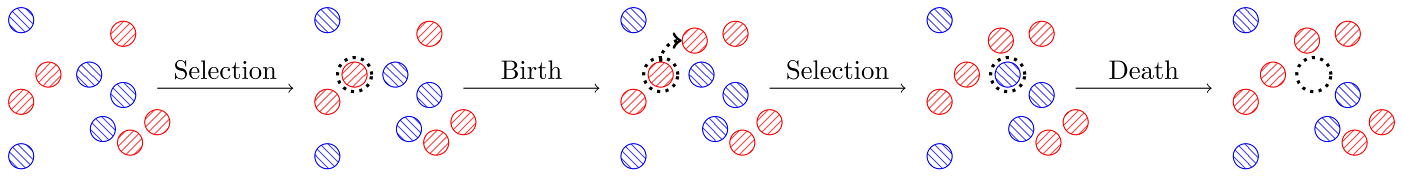


Figure 1: A diagrammatic representation of a Moran process.

The Moran process was initially introduced in [40]. It has since been used in a variety of settings including the understanding of the spread of cooperative and non-cooperative behaviour such as cancer [55] and the emergence of cooperative behaviour in spatial topologies [4]. However these works mainly consider relatively simple strategies. Some work has looked at evolutionary stability of agent-based strategies within the Prisoner's Dilemma [34] but this is not done in the more widely used setting of the Moran process, rather in terms of infinite population stability. In [14] Moran processes are studied in a theoretical framework for a small subset of strategies. The subset included memory one strategies: strategies that recall the events of the previous round only.

Of particular interest are the zero determinant strategies introduced in [44]. It was argued in [50] that generous ZD strategies are robust against invading strategies. However, in [32], a strategy using machine learning techniques was capable of resisting invasion and also able to invade any memory one strategy. Recent work [25] has investigated the effect of memory length on strategy performance and the emergence of cooperation but this is not done in a Moran process context and only considers specific cases of memory 2 strategies. In [1] it was recognised that many zero determinant strategies do not fare well against themselves. This is a disadvantage for the Moran process where the best strategies cooperate well with other players using the same strategy.

## 1.2 Strategies considered

To carry out this numerical experiment, 164 strategies, listed (with their properties) in Appendix A, are used from the Axelrod library. There are 43 stochastic and 121 deterministic strategies. Their memory depth, defined by the number of rounds of history used by the strategy each round, is shown in Table 1. The memory depth is infinite if the strategy uses the entire history of play (whatever its length). For example, a strategy that utilizes a handshaking mechanism where the opponent's actions on the first few rounds of play determines the strategies subsequent behavior would have infinite memory depth.

A number of these strategies have been trained with reinforcement learning algorithms prior to this study and not specifically for the Moran process.

- Evolved ANN: a neural network based strategy;
- Evolved LookerUp: a lookup table based strategy;
- PSO Gambler: a stochastic version of the lookup table based strategy;
- Evolved HMM: a hidden Markov model based strategy.

Apart from the PSO Gambler strategy, which was trained using a particle swarm optimisation algorithm, these strategies are trained with an evolutionary algorithm that perturbs strategy parameters and optimizes the mean total score against all other opponents [3]. They were trained to win IPD tournaments by maximizing their mean total payoffs against a variety of opponents. Variation is introduced via mutation and crossover of parameters, and the best performing strategies are carried to the next generation along with new variants. Similar methods appear in the literature [7].

More information about each player can be obtained in the documentation for [51] and a detailed description of the performance of these strategies in IPD tournaments will be described in upcoming manuscript(s).

All of the training code is archived at [23]. This software is (similarly to the Axelrod library) available on github `https://github.com/Axelrod-Python/axelrod-dojo` with documentation to train new strategies easily. Training typically takes less than 100 generations and can be completed within several hours on commodity hardware.

There are three further strategies trained specifically for this study; Trained FSM 1, 2, and 3 (TF1 - TF3). These are based on finite state machines of 16, 16, and 8 states respectively (see Figures 2, 3 and 4).

As opposed to the previously described strategies, these strategies were trained with the objective function of **mean fixation probabilities for Moran processes** starting at initial population states consisting of $N/2$ individuals of the training candidates and $N/2$ individuals of an opponent strategy, taken from a selection of 150 opponents from the Axelrod library:

- TF1 $N = 12$, 0% noise, 10000 repetitions per matchup
- TF2 $N = 10$, 0% noise, 10000 repetitions per matchup
- TF3 $N = 8$, 1% noise, 100 repetitions per matchup

Each matchup of players was run to fixation for the specified number of repetitions to estimate the absorption probabilities. The trained algorithms were run for fewer than 50 generations. Training data for this is available at [28].

TF3 cooperates and defects with various cycles depending on the opponent's actions. TF3 will mutually cooperate with any strategy and only tolerates a few defections before defecting for the rest of match. It is similar to but not exactly the same as Fool Me Once, a strategy that cooperates until the opponent has defected twice (not necessarily consecutively), and defects indefinitely thereafter. Though a product of training with a Moran objective, it differs from TF1 and TF2 in that it lacks a handshake mechanism. Figure 4 shows all 8 states of the strategy produced by the training process (states 3 and 8 are not reachable).

TF2 always starts with CD and will defect against opponents that start with DD. It plays CDD against itself and then cooperates thereafter; Fortress3 and Fortress4 also use a similar handshake and cooperate with TF2. Cooperation can be rescued after a failed handshake by a complex sequence of plays which sometimes results in mutual cooperation with Firm but Fair, Grofman, and GTFT, and a few others with low probability. TF2 defects against all other players in the study, barring unusual cases arising from particular randomizations. Figure 3 shows all 16 states of the strategy (states 6 and 7 are not reachable).

TF1 has an initial handshake of CCD and cooperates if the opponent matches. However if the opponent later defects, TF1 will respond in kind, so the handshake is not permanent. Only one player (Prober 4 [36]) manages to achieve cooperation with TF1 after about 20 rounds of play. TF1 is functionally very similar to a strategy known as "Collective

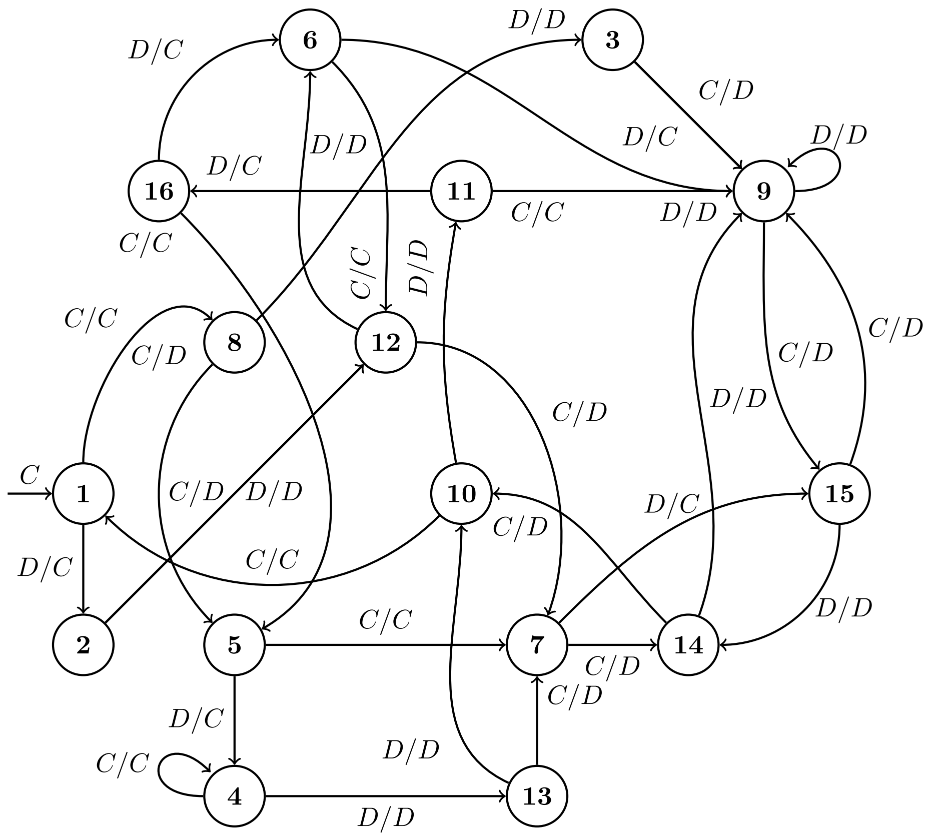


Figure 2: TF1: a 16 state finite state machine with a handshake leading to mutual cooperation at state 4.

Strategy", which has a handshake of CD and cooperates with opponents that matched the handshake until they defect, defecting thereafter if the opponent ever defects [33]. This strategy was specifically designed for evolutionary processes.

For both TF1 and TF2 a handshake mechanism naturally emerges from the structure of the underlying finite state machine. This behavior is an outcome of the evolutionary process and is in no way hard-coded or included via an additional mechanism.

| Memory Depth | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 9 | 10 | 11 | 12 | 16 | 20 | 40 | 200 | $\infty$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Count | 3 | 28 | 12 | 8 | 2 | 6 | 1 | 1 | 5 | 1 | 1 | 2 | 2 | 2 | 1 | 89 |

Table 1: Memory depth

### 1.3 Data collection

Each strategy pair is run for 1000 repetitions of the Moran process to fixation with starting population distributions of $(1, N-1)$, $(N/2, N/2)$ and $(N-1, 1)$, for $N$ from 2 through 14. The fixation probability is then empirically computed for each combination of starting distribution and value of $N$. The Axelrod library can carry out exact simulations of the Moran process. Since some of the strategies have a high computational cost or are stochastic, samples are taken from a large number of match outcomes for the pairs of players for use in computing fitnesses in the Moran process. This approach was verified to agree with unsampled calculations to a high degree of accuracy in specific cases. This is described in Algorithms 1 and 2.

Section 2 will further validate the methodology by comparing simulated results to analytical results in some cases. The main results of this manuscript are presented in Section 3 which will present a detailed analysis of all the data generated. Finally, Section 5 will conclude and offer future avenues for the work presented here.

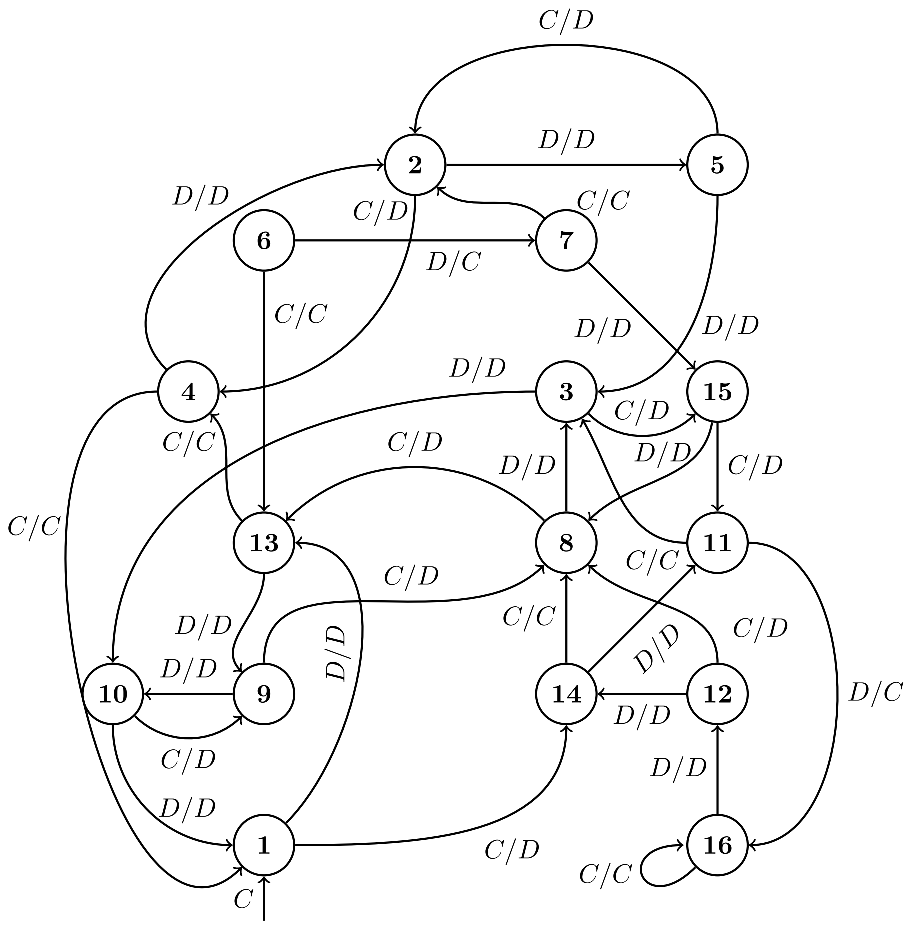


Figure 3: TF2: a 16 state finite state machine with a handshake leading to mutual cooperation at state 16.

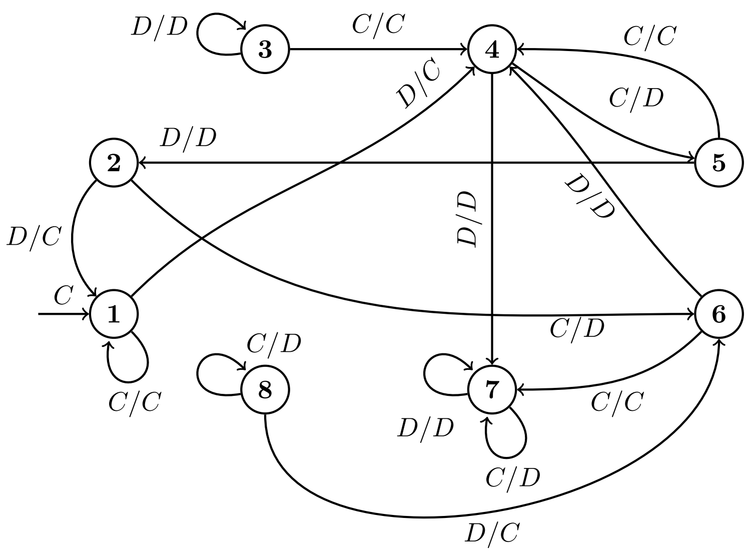


Figure 4: TF3: an 8 state finite state machine.

**Algorithm 1** Data Collection

```
for player one in players list do
    for player two in (players list - player one) do
        pair ← (player one, player two)
        for starting population distributions in $[(1, N-1), (\frac{N}{2}, \frac{N}{2}), (N-1, 1)]$ do
            while repetitions $\leq 1000$ do
                simulate moran process*(pair, starting distribution)
            end while
            return fixation probabilities
        end for
    end for
end for
```

**Algorithm 2** Moran process

```
initial population ← (pair, starting distribution)
population ← initial population
while population not uniform do
    for player in population do
        for opponent in (population - player) do
            match ← (player, opponent)
            results ← cache (match)
        end for
    end for
    population ← sorted(results)
    parent ← selected randomly in proportion to total match payoffs
    child ← parent
    kill off ← random player from population
    population ← child replaces kill off
end while
```

## 2 Validation

As described in [43] consider the payoff matrix:

$$M = \begin{pmatrix} a, b \\ c, d \end{pmatrix} \tag{1}$$

The expected payoffs of $i$ players of the first type in a population with $N - i$ players of the second type are given by:

$$f_i = \frac{a(i-1) + b(N-i)}{N-1} \tag{2}$$

$$g_i = \frac{ci + d(N-i-1)}{N-1} \tag{3}$$

The transitions within the birth death process that underpins the Moran process are then given by:

$$p_{i,i+1} = \frac{if_i}{if_i + (N-i)g_i}\frac{N-i}{N} \tag{4}$$

$$p_{i,i-1} = \frac{(N-i)g_i}{if_i + (N-i)g_i}\frac{i}{N} \tag{5}$$

$$p_{ii} = 1 - p_{i,i+1} - p_{i,i-1} \tag{6}$$

Using this it is a known result [4] that the fixation probability of the first strategy in a population of $i$ individuals of the first type and $N - i$ individuals of the second:

$$x_i = \frac{1 + \sum_{j=1}^{i-1}\prod_{k=1}^{j}\gamma_j}{1 + \sum_{j=1}^{N-1}\prod_{k=1}^{j}\gamma_j} \tag{7}$$

where:

$$\gamma_j = \frac{p_{j,j-1}}{p_{j,j+1}}$$

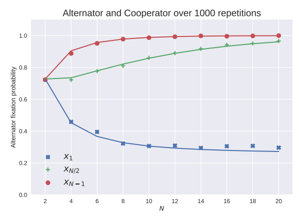
A neutral strategy will have fixation probability $x_i = i/N$.

Comparisons of $x_1, x_{N/2}, x_{N-1}$ are shown in Figure 5. The points represent the simulated values and the line shows the theoretical value. Note that these are all deterministic strategies and show a perfect match between the expected value of (7) and the actual Moran process for all strategy pairs. Figure 6 shows the fixation probabilities for stochastic strategies. These are no longer a good match which highlights the weakness of assuming a given interaction between two IPD strategies can be summarised with a set of utilities as shown in (1). For any given pair of strategies it is possible to obtain $p_{i,i-1}, p_{i,i+1}, p_{ii}$ exactly (as opposed to the approximations offered by (4), (5) and (6)). Obtaining these requires particular analysis for a given pair and can be quite a complex endeavour for stochastic strategies with long memory: this is not necessary for the purposes of this work. All data generated for this validation exercise can be found at [28].

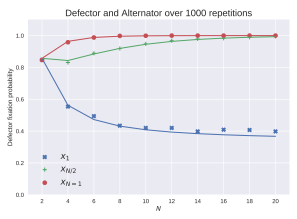
## 3 Empirical results

This section outlines the data analysis carried out:

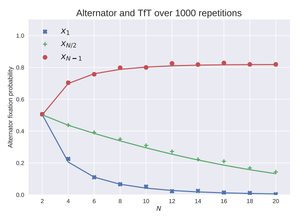
- Section 3.1 considers the specific case of $N = 2$.
- Section 3.2 investigates the effect of population size on the ability of a strategy to invade another population. This will highlight how complex strategies with long memories outperform simpler strategies.
- Section 3.3 similarly investigates the ability to defend against an invasion.
- Section 3.4 investigates the relationship between performance for differing population sizes as well as taking a close look at zero determinant strategies [44].



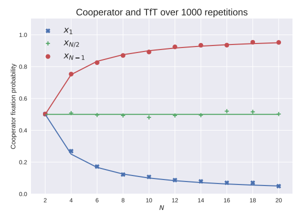
(a) Alternator and Cooperator



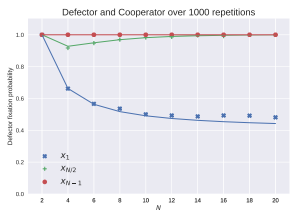
(b) Defector and Alternator



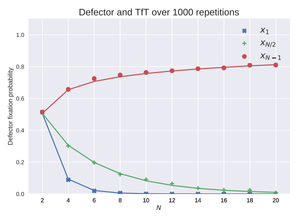
(c) Alternator and Tit For Tat



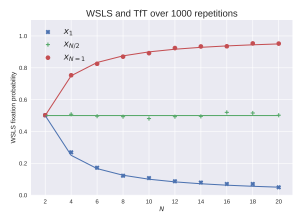
(d) Cooperator and Tit For Tat



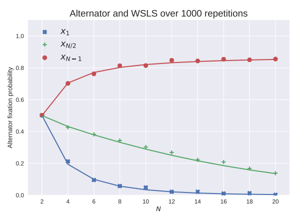
(e) Defector and Cooperator



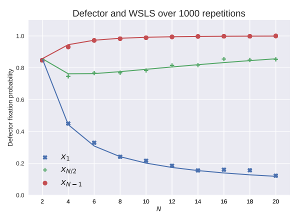
(f) Defector and Tit For Tat



(g) Win Stay Lose Shift and Tit For Tat



(h) Alternator and Win Stay Lose Shift



(i) Defector and Win Stay Lose Shift

Figure 5: Comparison of theoretic and actual Moran Process fixation probabilities for **deterministic** strategies.

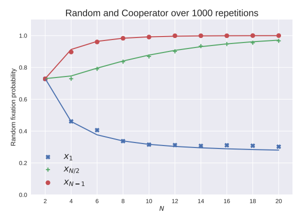
## 3.1 The special case of $N = 2$

When $N = 2$ the Moran process is effectively a measure of the distribution of relative mean payoffs over all possible matches between two players. The strategy that scores higher than the other more often will fixate more often. For $N = 2$ the two cases of $x_1$ and $x_{N-1}$ coincide, but will be considered separately for larger $N$ in sections 3.2 and 3.3. Figure 7a shows all fixation probabilities for the strategies considered. The top 16 (10%) strategies are shown in Table 7b. The top five ranking strategies are:

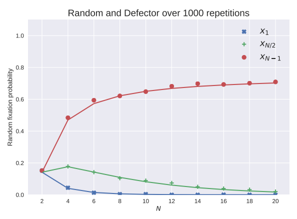
1. The top strategy is the Collective Strategy (CS) which has a simple handshake mechanism described above.
2. Defector: it always defects. Since it has no interactions with other defectors (recall that $N = 2$), its aggressiveness is rewarded.
3. Aggravater, which plays like Grudger (responding to any defections with unconditional defections throughout) however starts by playing 3 defections.
4. Predator, a finite state machine described in [7].
5. Handshake, a slightly less aggressive version of the Collective Strategy [46]. As long as the initial sequence is played then it cooperates. Thus it will do well in a population consisting of many members of itself just as the Collective Strategy does. The difference is that CS will defect after the handshake if the opponent defects while handshake will not.

It is also noted that TF1, TF2 and TF3 all perform well. This is also the $N$ for which a zero determinant strategy does appear in the top 10% ranking strategies: ZD-extort-4. The performance of zero determinant strategies will be examined more closely in Section 3.4.

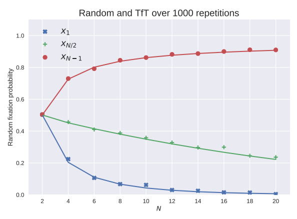
As will be demonstrated in Section 3.4 the results for $N = 2$ differ from those of larger $N$. Hence these results do not concur with the literature which suggests that zero determinant strategies should be effective for larger population sizes, but these analyses consider stationary behaviour, while this work runs for a fixed number of rounds. [50] The stationarity



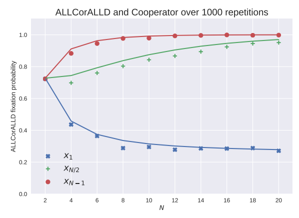
(a) Random and Cooperator



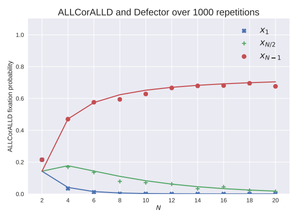
(b) Random and Defector



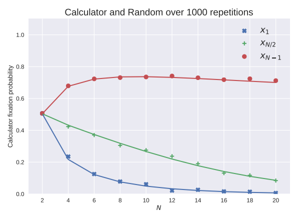
(c) Random and Tit For Tat



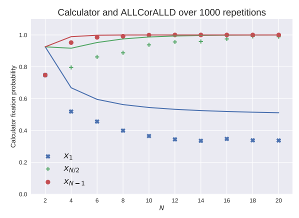
(d) All C or all D and Cooperator



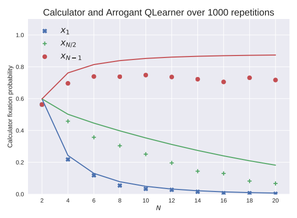
(e) All C or all D and Defector



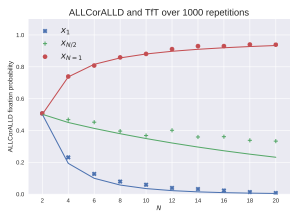
(f) Calculator and Random



(g) Calculator and All C or all D



(h) Calculator and Arrogant Q learner



(i) All C or all D and Tit For Tat

Figure 6: Comparison of theoretic and actual Moran Process fixation probabilities for **stochastic** strategies.

assumption allows for a deterministic payoff matrix leading to the conclusions about zero determinant strategies in the space of memory-one strategies that do not generalize to this context.

## 3.2 Strong Invaders

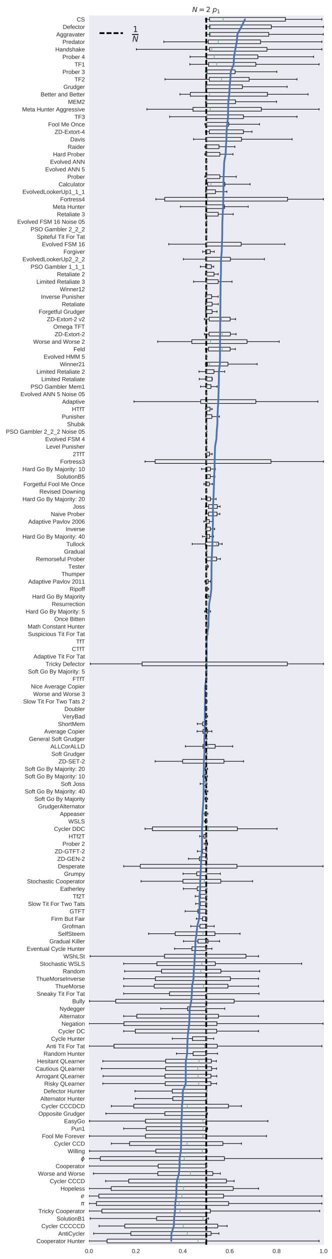
In this section the focus is on the ability of a mutant strategy to invade: the probability of one individual of a given type successfully fixating in a population of $N - 1$ other individuals, denoted by $x_1$. The ranks of each strategy for all considered values of $N$ according to mean $x_1$ are shown in Figure 8.

The fixation probabilities are shown in Figures 9a, 9b and 9c for $N \in \{3, 7, 14\}$ showing the mean fixation as well as the neutral fixation for each given scenario.

The top 16 strategies are given in Tables 2.

It can be seen that apart from CS, none of the strategies of Table 7b perform well for $N \in \{3, 7, 14\}$. The new top performing strategies are:

- Grudger (which only performs well for $N = 3$), starts by cooperating but will defect if at any point the opponent has defected.
- MEM2, an infinite memory strategy that switches between TFT, TF2T, and Defector [34].
- TF3, the finite state machine trained specifically for Moran processes described in Section 1.
- Prober 4, a strategy which starts with a specific 20 move sequence of cooperations and defections [36]. This initial sequence serves as approximate handshake.
- PSO Gambler and Evolved Lookerup 2 2 2: are strategies that make use of a lookup table mapping the first 2 moves of the opponent as well as the last 2 moves of both players to an action. The PSO gambler is a stochastic version of the Lookerup which maps those states to probabilities of cooperating. The Lookerup was described in [29].



(a) The fixation probabilities for $N = 2$

| | Player | Mean $p_1$ |
|---|---|---|
| 1 | CS | 0.6651 |
| 2 | Defector | 0.6496 |
| 3 | Aggravater | 0.6328 |
| 4 | Predator | 0.6301 |
| 5 | Handshake | 0.6240 |
| 6 | Prober 4 | 0.6183 |
| 7 | **TF1** | 0.6171 |
| 8 | Prober 3 | 0.6044 |
| 9 | **TF2** | 0.6026 |
| 10 | Grudger | 0.5996 |
| 11 | Better and Better | 0.5980 |
| 12 | MEM2 | 0.5942 |
| 13 | Meta Hunter Aggressive | 0.5933 |
| 14 | **TF3** | 0.5927 |
| 15 | Fool Me Once | 0.5892 |
| 16 | ZD-Extort-4 | 0.5867 |

(b) Top strategies for $N = 2$ (neutral fixation is $p = 0.5$)

Figure 7: Performance of strategies for $N = 2$.

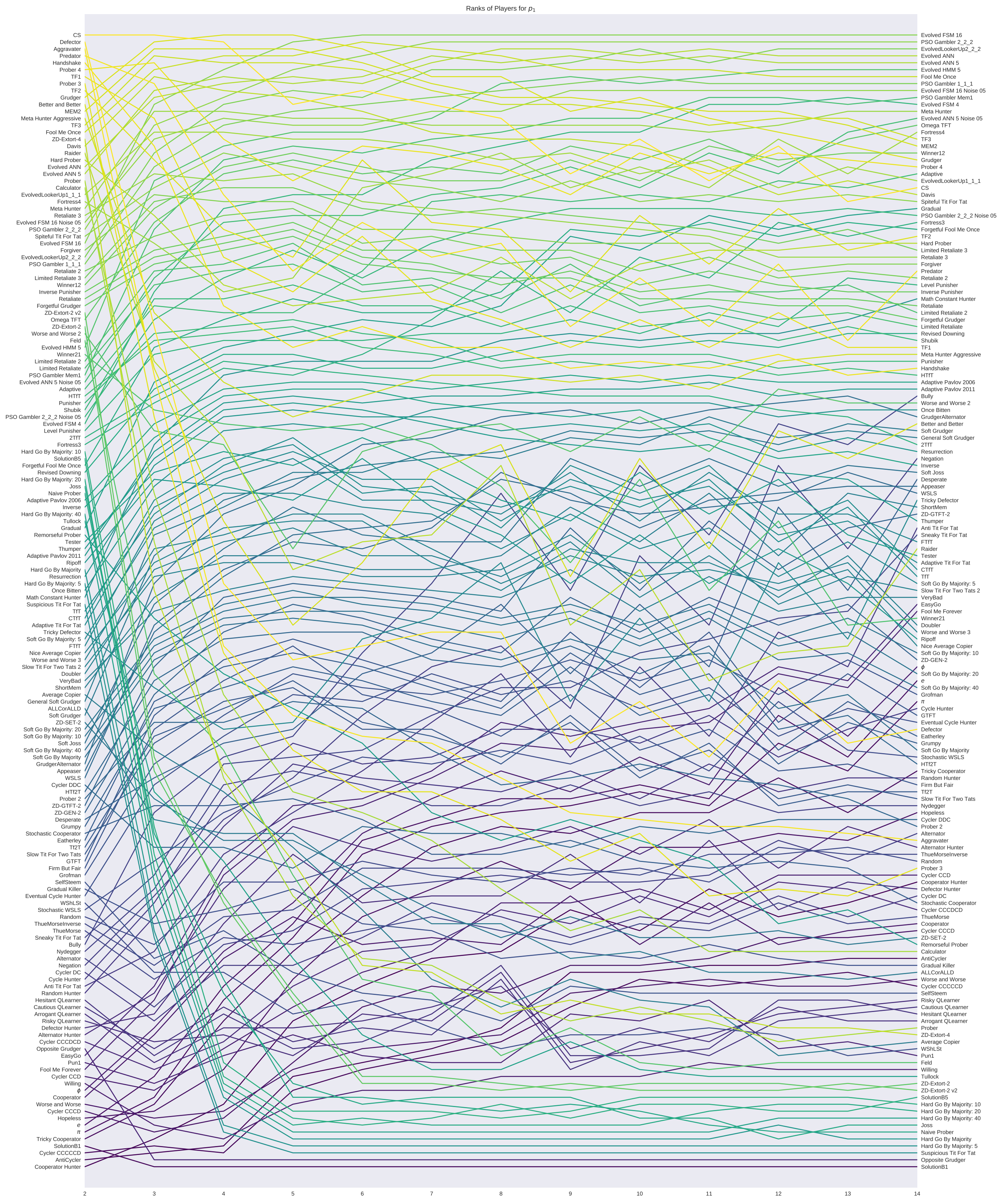
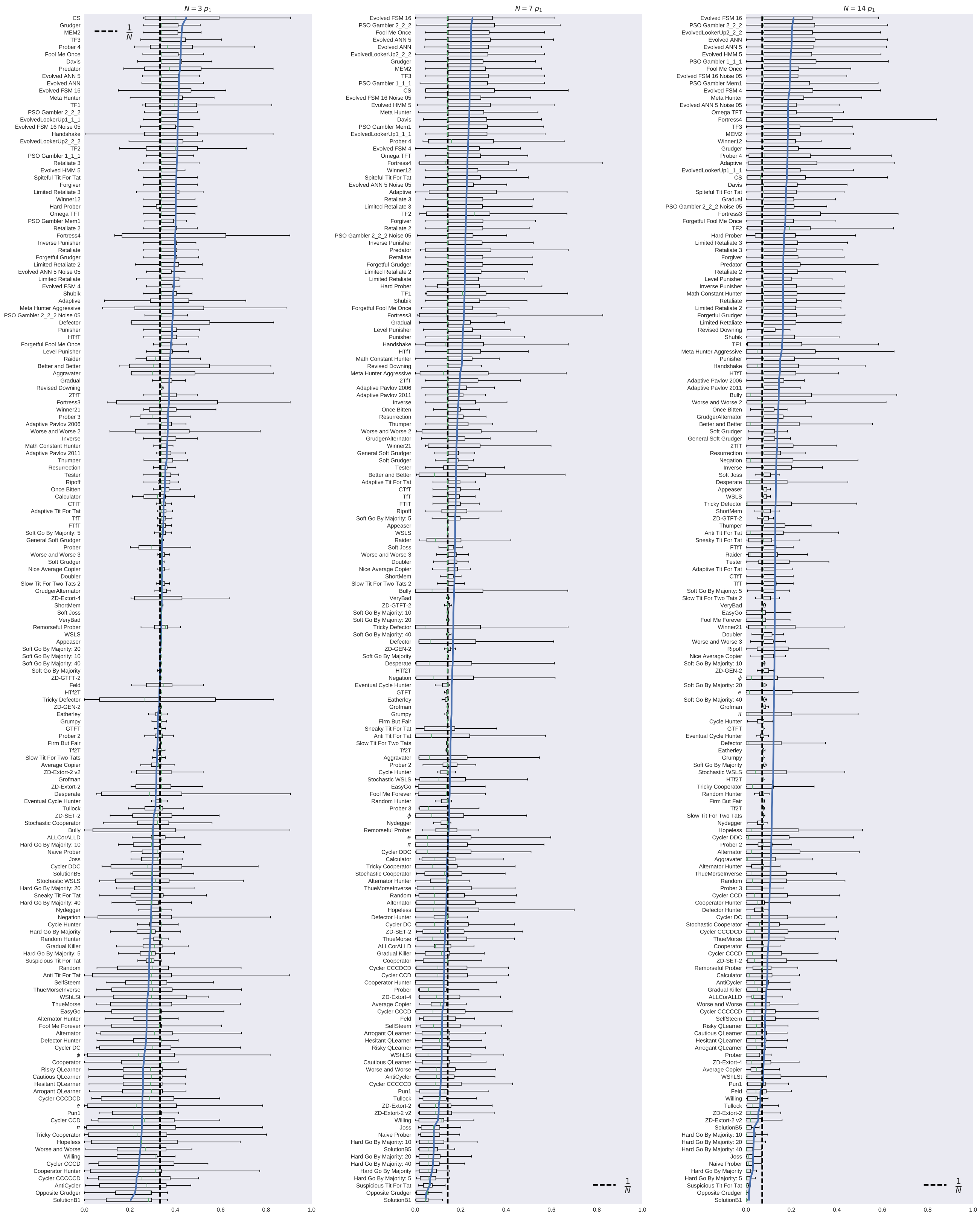


Figure 8: **Invasion**: Ranks of all strategies according to $x_1$ for different population sizes.



(a) $N = 3$ (b) $N = 7$ (c) $N = 14$

Figure 9: The fixation probabilities $x_1$.

| | Player | Mean $p_1$ |
|---|---|---|
| 1 | CS | 0.4478 |
| 2 | Grudger | 0.4313 |
| 3 | MEM2 | 0.4278 |
| 4 | **TF3** | 0.4267 |
| 5 | Prober 4 | 0.4242 |
| 6 | Fool Me Once | 0.4242 |
| 7 | Davis | 0.4218 |
| 8 | Predator | 0.4210 |
| 9 | Evolved ANN 5 | 0.4163 |
| 10 | Evolved ANN | 0.4163 |
| 11 | Evolved FSM 16 | 0.4154 |
| 12 | Meta Hunter | 0.4140 |
| 13 | **TF1** | 0.4139 |
| 14 | PSO Gambler 2_2_2 | 0.4134 |
| 15 | EvolvedLookerUp1_1_1 | 0.4113 |
| 16 | Evolved FSM 16 Noise 05 | 0.4107 |

(a) $N = 3$

| | Player | Mean $p_1$ |
|---|---|---|
| 1 | Evolved FSM 16 | 0.2523 |
| 2 | PSO Gambler 2_2_2 | 0.2467 |
| 3 | Fool Me Once | 0.2459 |
| 4 | Evolved ANN 5 | 0.2450 |
| 5 | Evolved ANN | 0.2449 |
| 6 | EvolvedLookerUp2_2_2 | 0.2443 |
| 7 | Grudger | 0.2442 |
| 8 | MEM2 | 0.2436 |
| 9 | **TF3** | 0.2430 |
| 10 | PSO Gambler 1_1_1 | 0.2404 |
| 11 | CS | 0.2395 |
| 12 | Evolved FSM 16 Noise 05 | 0.2394 |
| 13 | Evolved HMM 5 | 0.2390 |
| 14 | Meta Hunter | 0.2385 |
| 15 | Davis | 0.2379 |
| 16 | PSO Gambler Mem1 | 0.2348 |

(b) $N = 7$

| | Player | Mean $p_1$ |
|---|---|---|
| 1 | Evolved FSM 16 | 0.2096 |
| 2 | PSO Gambler 2_2_2 | 0.2042 |
| 3 | EvolvedLookerUp2_2_2 | 0.2014 |
| 4 | Evolved ANN | 0.2014 |
| 5 | Evolved ANN 5 | 0.2004 |
| 6 | Evolved HMM 5 | 0.1972 |
| 7 | PSO Gambler 1_1_1 | 0.1955 |
| 8 | Fool Me Once | 0.1955 |
| 9 | Evolved FSM 16 Noise 05 | 0.1943 |
| 10 | PSO Gambler Mem1 | 0.1920 |
| 11 | Evolved FSM 4 | 0.1918 |
| 12 | Meta Hunter | 0.1869 |
| 13 | Evolved ANN 5 Noise 05 | 0.1858 |
| 14 | Omega TFT | 0.1849 |
| 15 | Fortress4 | 0.1848 |
| 16 | **TF3** | 0.1846 |

(c) $N = 14$

Table 2: Top invaders for $N \in \{3, 7, 14\}$

- The Evolved ANN strategies are neural networks that map a number of attributes (first move, number of cooperations, last move, etc.) to an action. Both of these have been trained using an evolutionary algorithm.
- The Evolved FSM 16 is a 16 state finite state machine trained to perform well in tournaments.

Only one of the above strategies is stochastic although close inspection of the source code of PSO Gambler shows that it makes stochastic decisions rarely, and is functionally very similar to its deterministic cousin Evolved Looker Up. The PSO Gambler Mem1 strategy is a memory one strategy that has been trained to maximise its utility and does perform well. Apart from TF3, the finite state machines trained specifically for Moran processes do not appear in the top 5, while strategies trained for tournaments do. This is due to the nature of invasion: most of the opponents will initially be different strategies. The next section will consider the converse situation.

## 3.3 Strong resistors

In addition to identifying good invaders, strategies resistant to invasion by other strategies are identified by examining the distribution of $x_{N-1}$ for each strategy. The ranks of each strategy for all considered values of $N$ according to mean $x_{N-1}$ are shown in Figures 10.

The fixation probabilities are shown in Figures 11a, 9b and 11c for $N \in \{3, 7, 14\}$ showing the mean fixation as well as the neutral fixation for each given scenario.

Table 3 shows the top strategies when ranked according to $x_{N-1}$ for $N \in \{3, 7, 14\}$. Once again none of the short memory strategies from Section 3.1 perform well for high $N$.

| | Player | Mean $p_{N-1}$ |
|---|---|---|
| 1 | CS | 0.8359 |
| 2 | Predator | 0.8121 |
| 3 | **TF1** | 0.8087 |
| 4 | Handshake | 0.8014 |
| 5 | **TF2** | 0.7957 |
| 6 | Prober 4 | 0.7905 |
| 7 | Grudger | 0.7612 |
| 8 | Hard Prober | 0.7582 |
| 9 | **TF3** | 0.7570 |
| 10 | MEM2 | 0.7554 |
| 11 | Davis | 0.7536 |
| 12 | Winner21 | 0.7529 |
| 13 | Fool Me Once | 0.7489 |
| 14 | Fortress4 | 0.7467 |
| 15 | Retaliate 3 | 0.7448 |
| 16 | EvolvedLookerUp1_1_1 | 0.7422 |

(a) $N = 3$

| | Player | Mean $p_{N-1}$ |
|---|---|---|
| 1 | CS | 0.9765 |
| 2 | **TF1** | 0.9714 |
| 3 | **TF2** | 0.9677 |
| 4 | Predator | 0.9677 |
| 5 | Handshake | 0.9547 |
| 6 | Prober 4 | 0.9540 |
| 7 | Winner21 | 0.9392 |
| 8 | Hard Prober | 0.9331 |
| 9 | Fortress4 | 0.9255 |
| 10 | Grudger | 0.9198 |
| 11 | **TF3** | 0.9189 |
| 12 | Davis | 0.9186 |
| 13 | Ripoff | 0.9183 |
| 14 | Tester | 0.9176 |
| 15 | MEM2 | 0.9165 |
| 16 | Retaliate 3 | 0.9161 |

(b) $N = 7$

| | Player | Mean $p_{N-1}$ |
|---|---|---|
| 1 | CS | 0.9984 |
| 2 | **TF1** | 0.9973 |
| 3 | **TF2** | 0.9949 |
| 4 | Predator | 0.9941 |
| 5 | Prober 4 | 0.9863 |
| 6 | Handshake | 0.9812 |
| 7 | Winner21 | 0.9778 |
| 8 | Hard Prober | 0.9731 |
| 9 | Fortress4 | 0.9726 |
| 10 | Ripoff | 0.9669 |
| 11 | Tester | 0.9662 |
| 12 | Grudger | 0.9592 |
| 13 | **TF3** | 0.9589 |
| 14 | Davis | 0.9588 |
| 15 | Retaliate 3 | 0.9580 |
| 16 | Retaliate | 0.9576 |

(c) $N = 14$

Table 3: Top resistors for $N \in \{3, 7, 14\}$

Interestingly none of these strategies is stochastic: this is explained by the need of strategies to have a steady hand when interacting with their own kind. Acting stochastically increases the chance of friendly fire. However it is possible to design a strategy with a stochastic or error-correcting handshake that is an excellent resistor even in noisy environments [32].

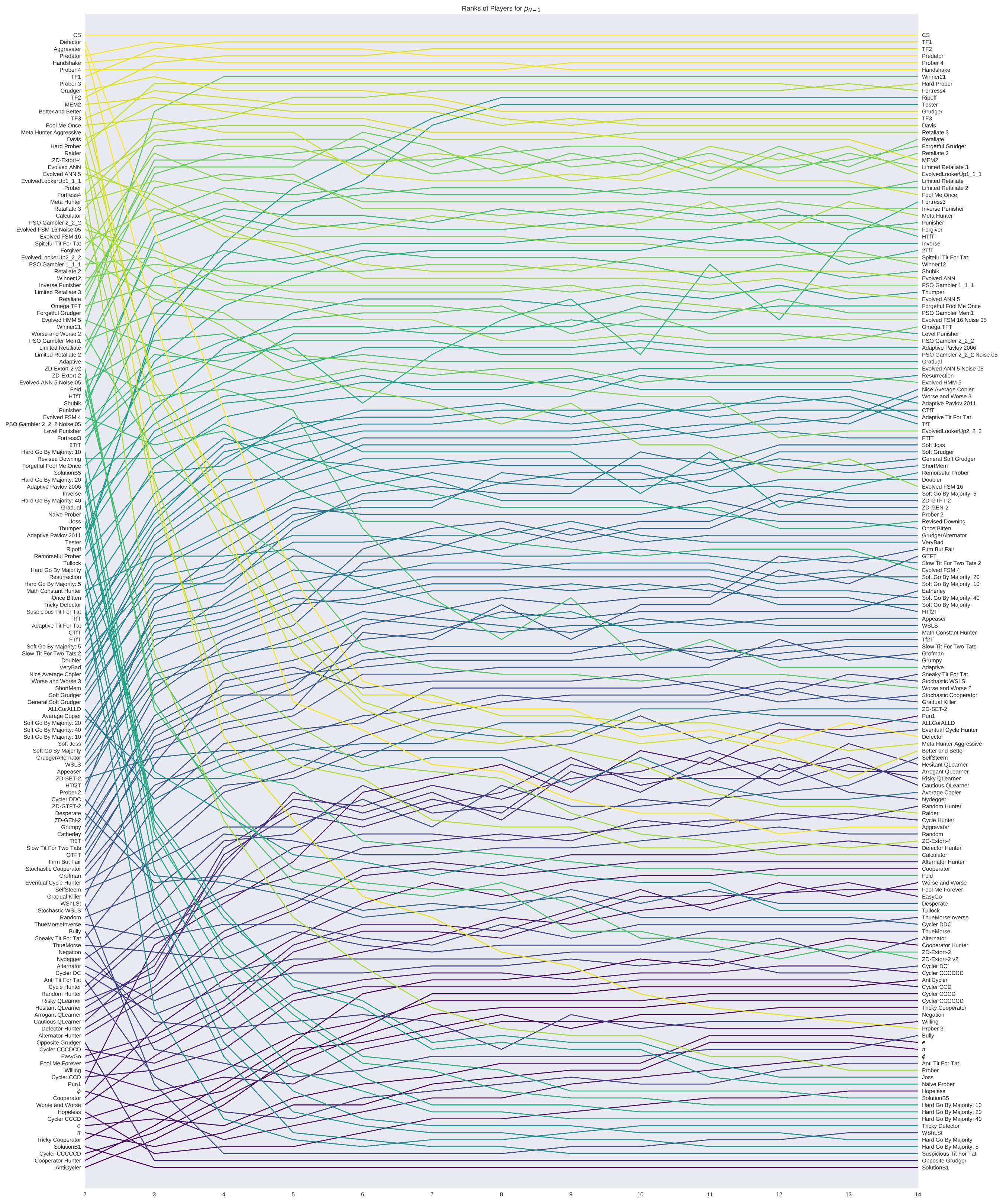
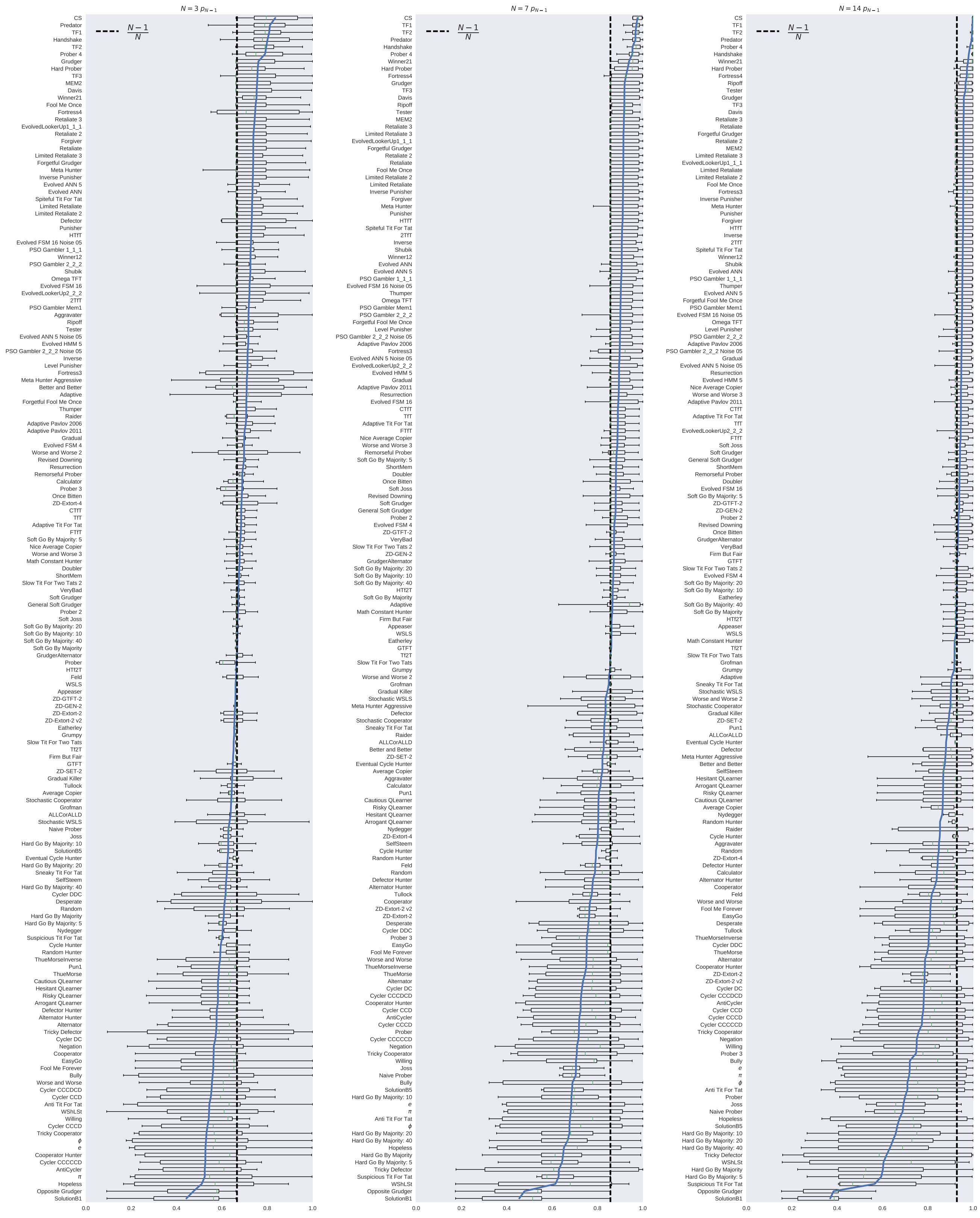


Figure 10: **Resistance**: Ranks of all strategies according to $x_{N-1}$ for different population sizes.



(a) $N = 3$ (b) $N = 7$ (c) $N = 14$

Figure 11: The fixation probabilities $x_{N-1}$.

There are are only two new strategies that appear in the top ranks for $x_{N-1}$: TF1 and TF2. These two strategies are with CS the strongest resistors. They all have handshakes, and whilst the handshakes of CS and Handshake (which ranks highly for the smaller values of $N$) were programmed, the handshakes of TF1 and TF2 evolved through an evolutionary process without any priming.

As described in Section 3.2 the strategies trained with the payoff maximizing objective are among the best invaders in the library however they are not as resistant to invasion as the strategies trained using a Moran objective function. These strategies include trained finite state machine strategies, but they do not appear to have handshaking mechanisms. Therefore it is reasonable to conclude that the objective function is the cause of the emergence of handshaking mechanisms. More specifically, TF1 and TF2 evolved handshakes for high invasion resistance. TF3 is a better total payoff maximizer which makes it a better invader along with the strategies trained to maximize total payoff since successful fitness proportionate selection is necessary for invasion. Training with an objective with initial population mix other than $(N/2, N/2)$ may favor invasion or resistance.

The payoff maximizing strategies typically will not defect before the opponent's first defection, possibly because the training strategy collection contains some strategies such as Grudger and Fool Me Once that retaliate harshly by defecting for the remainder of the match if the opponent has more than a small number of cumulative defections. Paradoxically it is advantageous to defect (as a signal) in order to achieve mutual cooperation with opponents using the same strategy but not with other opponents. Nevertheless an evolutionary process is able to tunnel through the costs and risks associated to early defections to find more optimal solutions, so it is not surprising in hindsight that handshaking strategies emerge from the evolutionary training process.

A handshake requires at least one defection and there is selective pressure to defect as few times as possible to achieve the self-recognition mechanism. It is also unwise to defect on the first move as some strategies additionally retaliate first round defections. So the handshakes used by TF1, TF2, and CS are in some sense optimal.

It is evident through Sections 3.1, 3.2 and 3.3 that performance of strategies not only depends on the initial population distribution but also that there seems to be a difference depending on whether or not $N > 2$. This will be explored further in the next section, looking not only at $x_1$ and $x_{N-1}$ but also consider $x_{N/2}$.

## 3.4 The effect of population size

To complement Figures 8 and 10, Figure 12 shows the rank of each strategy based on $x_{N/2}$. Tables 4, 5 and 6 show the same information for a selection of strategies:

- The strategies that ranked highly for $N = 2$;
- The strategies that ranked highly for $N = 14$;
- The zero determinant strategies.

The results for $x_{N/2}$ show similarities to the results for $x_{N-1}$ and in particular TF1, TF2 and TF3 ranked one, three and eight. This is to be expected since, as described in Section 1.2 these strategies were trained in an initial population of $(N/2, N/2)$ individuals.

For all starting populations $i \in \{1, N/2, N-1\}$ the ranks of strategies are relatively stable across the different values of $N > 2$ however for $N = 2$ there is a distinct difference. This highlights that there is little that can be inferred about the evolutionary performance of a strategy in a large population from its performance in a small population. This is confirmed by the performance of the zero determinant strategies: while some do rank relatively highly for $N = 2$ (ZD-extort-4 has rank 16) this rank does not translate to larger populations.

Figure 13 show the correlation coefficients of the ranks of strategies in differing population size. How well a strategy performs in any Moran process for $N > 2$ has little to do with the performance for $N = 2$. This illustrates why the strong performance of zero determinant strategies predicted in [44] does not extend to larger populations. This was discussed theoretically in [1] and observed empirically in these simulations.

# 4 Discussion

Training strategies to excel at the Moran process leads to the evolution of cooperation, but only with like individuals in the case of TF1 and TF2. This may have significant implications for human social interactions such as the evolution of ingroup/outgroup mechanisms and other sometimes costly rituals that reinforce group behavior.

While TF1 and TF2 are competent invaders, the best invaders in the study do not appear to employ strict handshakes, and are generally cooperative strategies. TF3, which does not use a handshake, is a better invader than TF1 and TF2

| Size | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CS | 1 | 1 | 2 | 11 | 9 | 11 | 13 | 21 | 16 | 22 | 17 | 25 | 23 |
| Defector | 2 | 43 | 80 | 91 | 89 | 87 | 87 | 103 | 97 | 105 | 94 | 103 | 101 |
| Aggravater | 3 | 50 | 89 | 99 | 102 | 103 | 108 | 113 | 114 | 115 | 115 | 116 | 117 |
| Predator | 4 | 8 | 24 | 35 | 28 | 33 | 31 | 43 | 36 | 43 | 34 | 45 | 35 |
| Handshake | 5 | 17 | 40 | 46 | 43 | 46 | 46 | 49 | 48 | 49 | 47 | 50 | 49 |
| Evolved FSM 16 | 31 | 11 | 6 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| PSO Gambler 2_2_2 | 29 | 14 | 10 | 6 | 4 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| EvolvedLookerUp2_2_2 | 33 | 18 | 11 | 9 | 10 | 6 | 6 | 5 | 3 | 5 | 3 | 3 | 3 |
| Evolved ANN | 20 | 10 | 8 | 7 | 8 | 5 | 3 | 3 | 4 | 3 | 4 | 4 | 4 |
| Evolved ANN 5 | 21 | 9 | 7 | 8 | 7 | 4 | 5 | 4 | 5 | 4 | 5 | 5 | 5 |
| **TF1** | 7 | 13 | 33 | 38 | 30 | 39 | 42 | 46 | 42 | 46 | 41 | 46 | 46 |
| **TF2** | 9 | 19 | 29 | 33 | 19 | 28 | 29 | 38 | 27 | 34 | 26 | 32 | 30 |
| **TF3** | 14 | 4 | 5 | 5 | 6 | 9 | 11 | 11 | 12 | 14 | 13 | 13 | 16 |
| ZD-Extort-4 | 16 | 81 | 107 | 120 | 135 | 136 | 142 | 140 | 142 | 142 | 144 | 144 | 145 |
| ZD-Extort-2 v2 | 41 | 105 | 126 | 140 | 152 | 152 | 153 | 152 | 153 | 153 | 153 | 152 | 153 |
| ZD-Extort-2 | 43 | 107 | 125 | 139 | 151 | 151 | 152 | 153 | 152 | 152 | 152 | 153 | 152 |
| ZD-SET-2 | 100 | 111 | 117 | 117 | 122 | 127 | 131 | 128 | 131 | 131 | 130 | 132 | 131 |
| ZD-GTFT-2 | 112 | 92 | 82 | 80 | 81 | 82 | 84 | 72 | 81 | 71 | 78 | 72 | 70 |
| ZD-GEN-2 | 113 | 96 | 87 | 83 | 85 | 88 | 90 | 82 | 87 | 82 | 86 | 83 | 91 |

Table 4: Invasion: Fixation ranks of some strategies according to $x_1$ for different population sizes

| Size | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CS | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Defector | 2 | 29 | 55 | 79 | 94 | 97 | 98 | 98 | 102 | 101 | 103 | 100 | 102 |
| Aggravater | 3 | 42 | 71 | 97 | 101 | 106 | 107 | 111 | 113 | 113 | 116 | 115 | 115 |
| Predator | 4 | 2 | 3 | 3 | 3 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| Handshake | 5 | 4 | 5 | 5 | 5 | 5 | 5 | 6 | 6 | 6 | 6 | 6 | 6 |
| **TF1** | 7 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| **TF2** | 10 | 5 | 4 | 4 | 4 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| Prober 4 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 5 | 5 | 5 | 5 | 5 | 5 |
| **TF3** | 13 | 9 | 10 | 11 | 11 | 11 | 13 | 14 | 13 | 13 | 13 | 13 | 13 |
| ZD-Extort-4 | 19 | 68 | 98 | 106 | 108 | 114 | 115 | 115 | 118 | 118 | 117 | 118 | 117 |
| ZD-Extort-2 v2 | 49 | 98 | 111 | 121 | 123 | 124 | 124 | 130 | 130 | 132 | 134 | 132 | 134 |
| ZD-Extort-2 | 50 | 97 | 112 | 123 | 124 | 125 | 123 | 126 | 131 | 131 | 132 | 133 | 133 |
| ZD-SET-2 | 108 | 105 | 104 | 104 | 103 | 103 | 100 | 100 | 101 | 99 | 98 | 98 | 98 |
| ZD-GTFT-2 | 112 | 95 | 88 | 84 | 75 | 72 | 71 | 73 | 71 | 71 | 67 | 68 | 68 |
| ZD-GEN-2 | 114 | 96 | 89 | 86 | 77 | 75 | 72 | 74 | 72 | 72 | 68 | 69 | 69 |

Table 5: Resistance: Fixation ranks of some strategies according to $x_{N-1}$ for different population sizes

| Size | 2 | 4 | 6 | 8 | 10 | 12 | 14 |
|---|---|---|---|---|---|---|---|
| CS | 1 | 1 | 1 | 1 | 1 | 1 | 2 |
| Defector | 2 | 78 | 99 | 106 | 110 | 113 | 120 |
| Aggravater | 3 | 91 | 105 | 111 | 122 | 125 | 128 |
| Predator | 4 | 2 | 4 | 4 | 4 | 4 | 4 |
| Handshake | 5 | 6 | 5 | 6 | 6 | 6 | 6 |
| **TF2** | 9 | 4 | 3 | 2 | 2 | 2 | 1 |
| **TF1** | 7 | 3 | 2 | 3 | 3 | 3 | 3 |
| Prober 4 | 6 | 5 | 6 | 5 | 5 | 5 | 5 |
| **TF3** | 14 | 8 | 8 | 8 | 8 | 8 | 8 |
| ZD-Extort-4 | 16 | 102 | 117 | 129 | 141 | 143 | 145 |
| ZD-Extort-2 v2 | 41 | 118 | 135 | 151 | 152 | 152 | 153 |
| ZD-Extort-2 | 43 | 117 | 136 | 149 | 151 | 151 | 152 |
| ZD-SET-2 | 100 | 110 | 110 | 108 | 106 | 106 | 108 |
| ZD-GTFT-2 | 112 | 82 | 80 | 77 | 75 | 75 | 74 |
| ZD-GEN-2 | 113 | 85 | 81 | 82 | 79 | 77 | 76 |

Table 6: Ranks of some strategies according to $x_{N/2}$ for different population sizes

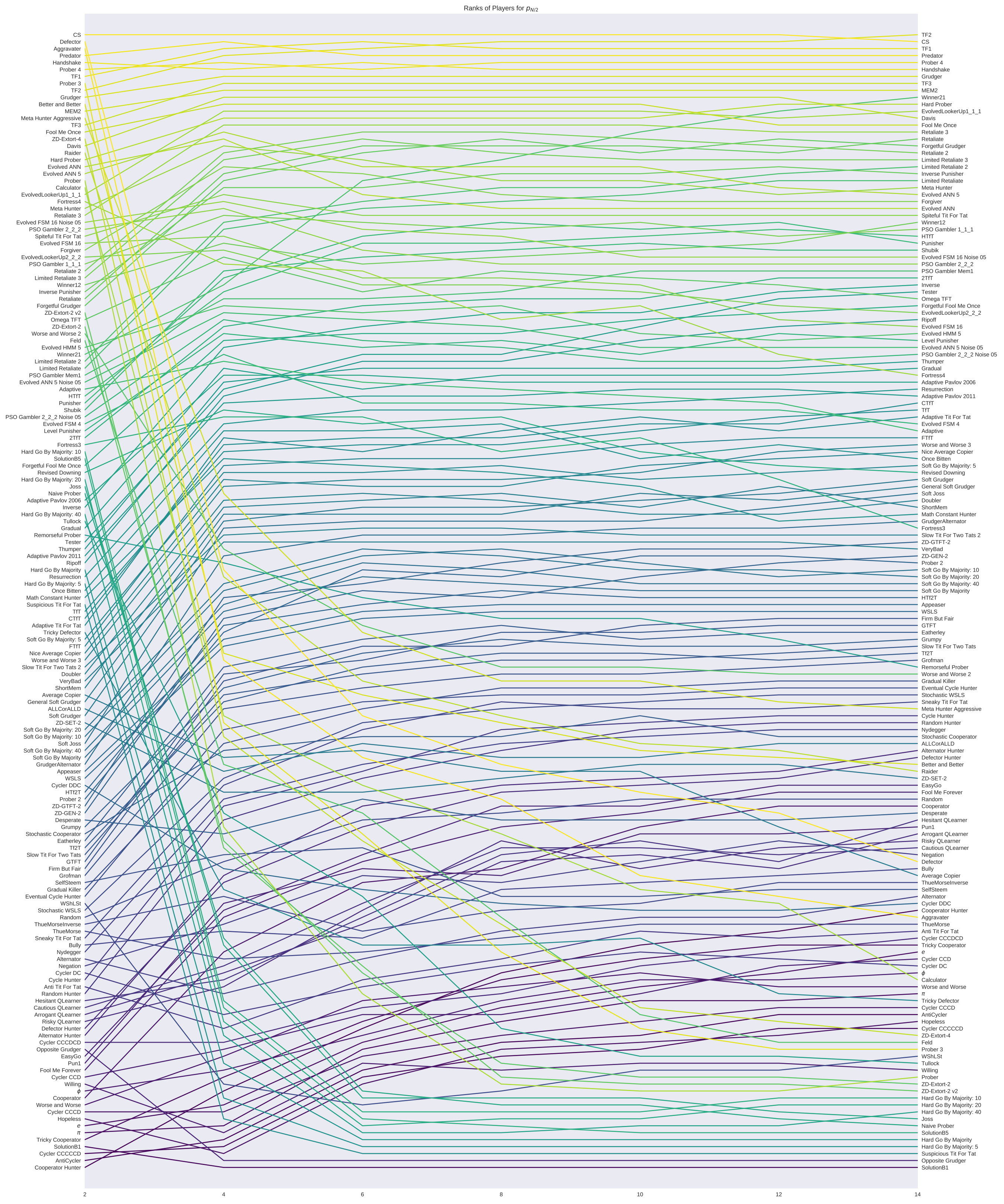
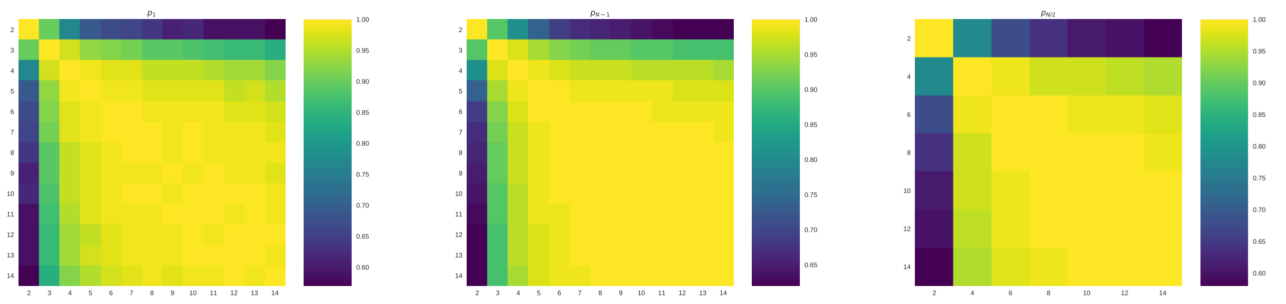


Figure 12: Fixation ranks of all strategies according to $x_{N/2}$ for different population sizes.



(a) Rank based on $x_1$ (b) Rank based on $x_{N-1}$ (c) Rank based on $x_{N/2}$

Figure 13: Heatmap of correlation coefficients of rankings by population size.

but not as good a resistor. Nevertheless it was the result of the same kind of training procceses and is a better combined invader-resistor than the invaders that were trained previously to maximize payout.

The strategies trained to maximize payoff in head-to-head matches are generally cooperative and are effective invaders. Combined with the fact that handshaking strategies are stronger resisters, this suggests that while maximizing individual payoff can lead to the evolution of cooperation, these strategies are not the most evolutionarily stable in the long run. A strategy with a handshaking mechanism is still capable of invading and is more resistant to subsequent invasions. Moreover, the best resistor of the payoff maximally trained strategies (Evolved Looker Up 1_1_1), which always defects if the opponent defects in the first round, is effectively employing a one-shot handshake of C. These insights also suggest that a strategy aware of the population distribution could choose to become a handshaker at a critical threshold and use a different strategy for invasion when in the minority. Information about the population distribution was not available to our strategies.

We did not attempt other objective functions that may serve to select for both invasion and resistance better than training at a starting population of $(N/2, N/2)$. Nevertheless our results suggest that there is not much room for improvement. Any handshake more sophisticated than always cooperate necessarily involves a defection. (A handshake consisting of a long sequence of cooperations is effectively a grudger.) For TF3 or EvolvedLookerUp1_1_1 to become better resistors they need a longer or more strict handshake. But if this handshake involves a defection then likely the invasion ability is diminished for $N > 2$: the top invaders for larger $N$ are nice strategies that do not defect before their opponents. This is because good invaders need to maximize match payoff to benefit from fitness proportionate selection, and so in the absence of a handshake mechanism, knowledge of the population distribution, or some identifying label on the opponent, a strategy must be generally cooperative. Aggressive strategies are only effective invaders for the smallest $N$, dropping dramatically as the population size increases.

We did, however, attempt to evolve CS using FSM and lookup table based players, which resulted in some very similar strategies. In particular we evolved a lookup strategy that had a handshake of DC and played TFT with other players after a correct handshake while defecting otherwise, which is quite close in function to CS (full grudging is not possible with a lookup table).

Finally we note that it may be possible to achieve similar results with smaller capacity finite state machine players.

## 5 Conclusion

A detailed empirical analysis of 164 strategies of the IPD within a pairwise Moran process has been carried out. All $\binom{164}{2} = 13,366$ possible ordered pairs of strategies have been placed in a Moran process with different starting values allowing the each strategy to attempt to invade the other. This is the largest such experiment carried out and has lead to many insights.

When studying evolutionary processes it is vital to consider $N > 2$ since results for $N = 2$ cannot be used to extrapolate performance in larger populations. This was shown both observationally in Sections 3.2 and 3.3 but also by considering the correlation of the ranks in different population sizes in Section 3.4.

Memory one strategies do not perform well in general in this study. There are no memory one strategies in the top 5 performing strategies for $N > 3$. This is due at least partly to their lack of ability to recognise their opponents. More sophisticated strategies prove to be high performers for invasion: these are infinite memory strategies which have been trained using a number of reinforcement learning algorithms. Interestingly they have been trained to perform well in tournaments and not Moran processes which highlights the potential for improvement.

One of the major findings discussed in Section 3.3, is the ability of strategies with a handshake mechanism to resist invasion. This was not only revealed for CS (a human designed strategy) but also for two FSM strategies (TF1 and TF2) specifically trained through an evolutionary process. In these two cases, the handshake mechanism was a product of the evolutionary process. Figure 14 shows the cooperation rate of TF1, TF2, TF3 and CS for each round of a match against all the opponents in this study. While TF3 does not have a strict handshake mechanism it is clear that all these strategies start a match by cooperating. It is then evident that TF3 cooperates more than the other strategies thus explaining the difference in performance. It is also clear that CS only cooperates with itself and Handshake: it is a very aggressive strategy.

These findings are important for the ongoing understanding of population dynamics and offer evidence for some of the shortcomings of low memory which has started to be recognised by the community [25].

All source code for this work has been written in a sustainable manner: it is open source, under version control and tested which ensures that all results can be reproduced [45, 48, 56]. The raw data as well as the processed data has also been properly archived and can be found at [28].

There are many opportunities to build on this work. In particular, an analysis of the effect of noise should offer insights regarding the stability of the findings, particularly for the handshaking strategies. They may be less dominant for larger amounts of noise since the handshaking mechanisms may become brittle. There are many other variations to explore including populations with more than one type, spatial structure, and mutation.

## Acknowledgements

This work was performed using the computational facilities of the Advanced Research Computing @ Cardiff (ARCCA) Division, Cardiff University.

A variety of software libraries have been used in this work:

- The Axelrod library (IPD strategies and Moran processes) [51].
- The matplotlib library (visualisation) [26].
- The pandas and numpy libraries (data manipulation) [38, 54].

## References

[1] Christoph Adami and Arend Hintze. "Evolutionary instability of zero-determinant strategies demonstrates that winning is not everything." In: *Nature communications* 4.1 (2013), p. 2193. ISSN: 2041-1723. DOI: 10.1038/ncomms3193. arXiv: arXiv:1208.2666v4. URL: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3741637%7B%5C&%7Dtool=pmcentrez%7B%5C&%7Drendertype=abstract.

[2] Christoph Adami, Jory Schossau, and Arend Hintze. "Evolutionary game theory using agent-based methods". In: *Physics of life reviews* 19 (2016), pp. 1–26.

[3] Michael Affenzeller et al. *Genetic algorithms and genetic programming: modern concepts and practical applications.* Crc Press, 2009.

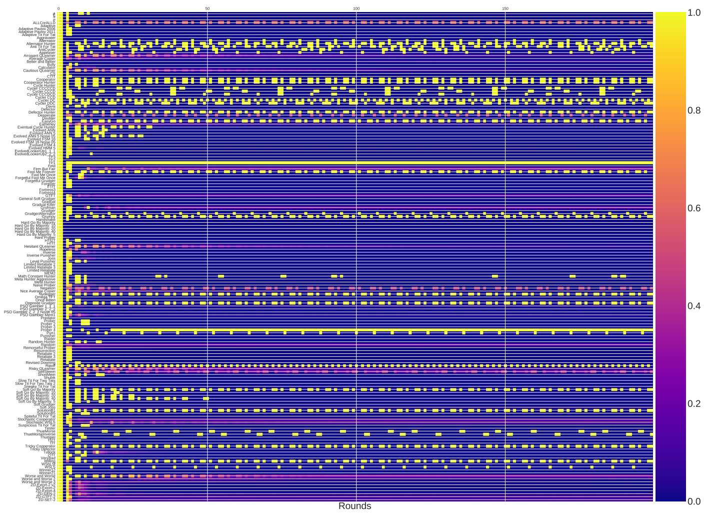
[4] B. Allen et al. "Evolutionary dynamics on any population structure". In: 544 (Mar. 2017), pp. 227–230. DOI: 10.1038/nature21723. arXiv: 1605.06530 [q-bio.PE].

[5] Daniel Ashlock, Joseph Alexander Brown, and Philip Hingston. "Multiple Opponent Optimization of Prisoners Dilemma Playing Agents". In: *IEEE Transactions on Computational Intelligence and AI in Games* 7.1 (2015), pp. 53–65.

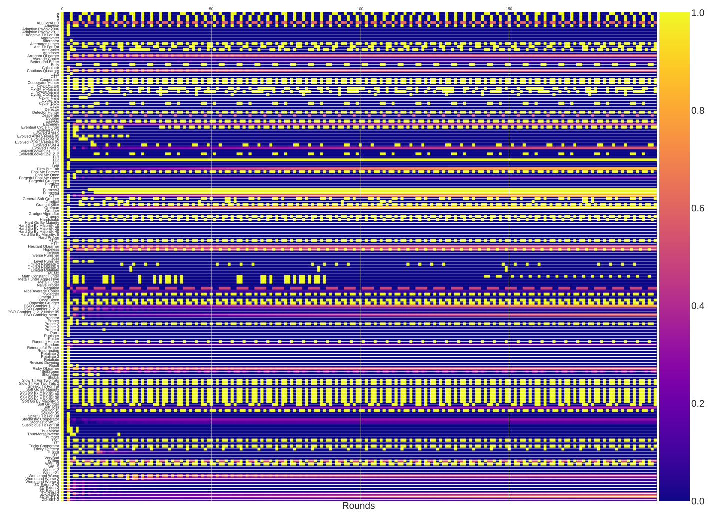
[6] Daniel Ashlock and Eun-Youn Kim. "Fingerprinting: Visualization and automatic analysis of prisoner's dilemma strategies". In: *IEEE Transactions on Evolutionary Computation* 12.5 (2008), pp. 647–659.

[7] Wendy Ashlock and Daniel Ashlock. "Changes in Prisoner ' s Dilemma Strategies Over Evolutionary Time With Different Population Sizes". In: (2006), pp. 1001–1008.

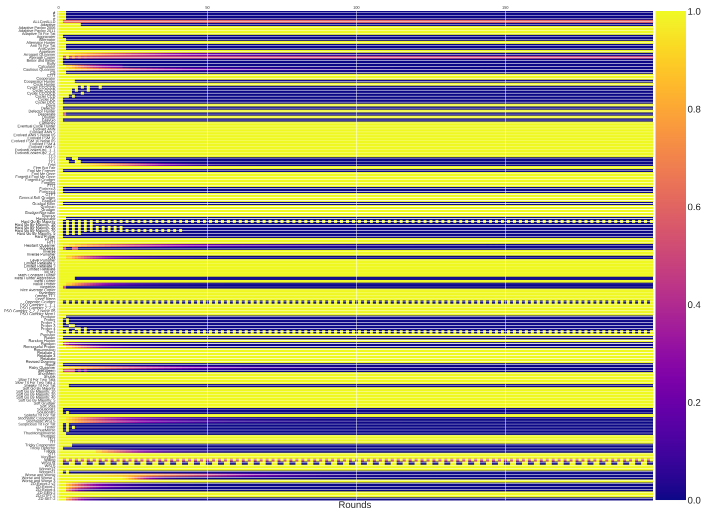
[8] Wendy Ashlock and Daniel Ashlock. "Changes in prisoners dilemma strategies over evolutionary time with different population sizes". In: *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on.* IEEE. 2006, pp. 297–304.



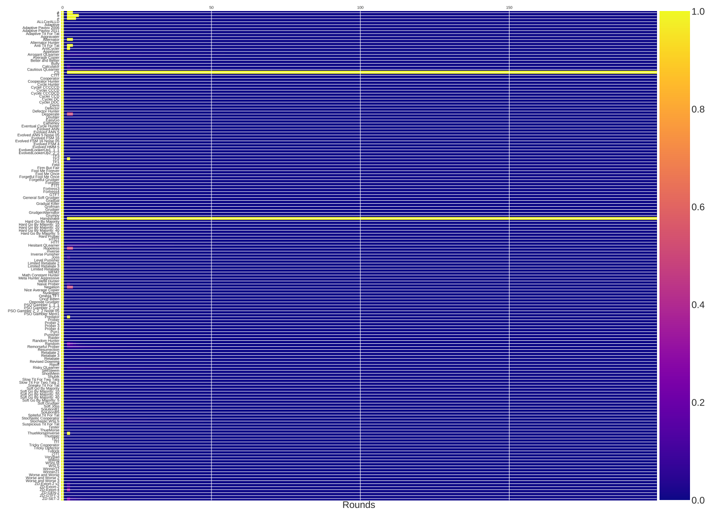
(a) TF1



(b) TF2



(c) TF3



(d) CS

Figure 14: Cooperation rate per round (over 10000 repetitions).

[9] Wendy Ashlock, Jeffrey Tsang, and Daniel Ashlock. “The evolution of exploitation”. In: *Foundations of Computational Intelligence (FOCI), 2014 IEEE Symposium on*. IEEE. 2014, pp. 135–142.

[10] R. Axelrod. “Effective Choice in the Prisoner’s Dilemma”. In: *Journal of Conflict Resolution* 24.1 (1980), pp. 3–25.

[11] R. Axelrod. “More Effective Choice in the Prisoner’s Dilemma”. In: *Journal of Conflict Resolution* 24.3 (1980), pp. 379–403. ISSN: 0022-0027. DOI: 10.1177/002200278002400301.

[12] Robert Axelrod. “Effective choice in the prisoner’s dilemma”. In: *Journal of conflict resolution* 24.1 (1980), pp. 3–25.

[13] Robert M Axelrod. *The evolution of cooperation*. Basic books, 2006.

[14] Seung Ki Baek et al. “Comparing reactive and memory- one strategies of direct reciprocity”. In: *Nature Publishing Group* (2016), pp. 1–13. DOI: 10.1038/srep25676. URL: http://dx.doi.org/10.1038/srep25676.

[15] Jeffrey S Banks and Rangarajan K Sundaram. “Repeated games, finite automata, and complexity”. In: *Games and Economic Behavior* 2.2 (1990), pp. 97–117.

[16] Bruno Beaufils, Jean-Paul Delahaye, and Philippe Mathieu. “Our meeting with gradual, a good strategy for the iterated prisoners dilemma”. In: *Proceedings of the Fifth International Workshop on the Synthesis and Simulation of Living Systems*. 1997, pp. 202–209.

[17] Pieter van den Berg and Franz J Weissing. “The importance of mechanisms for the evolution of cooperation”. In: *Proc. R. Soc. B*. Vol. 282. 1813. The Royal Society. 2015, p. 20151382.

[18] Andre LC Carvalho et al. “Iterated Prisoners Dilemma-An extended analysis”. In: (2013).

[19] Eckhart Arnold. *CoopSim v0.9.9 beta 6*. 2015. URL: https://github.com/jecki/CoopSim/.

[20] Merrill M. Flood. *Some Experimental Games*. 1958. DOI: 10.1287/mnsc.5.1.5.

[21] Marcus R Frean. “The prisoner’s dilemma without synchrony”. In: *Proceedings of the Royal Society of London B: Biological Sciences* 257.1348 (1994), pp. 75–79.

[22] Marco Gaudesi et al. “Exploiting evolutionary modeling to prevail in iterated prisoners dilemma tournaments”. In: *IEEE Transactions on Computational Intelligence and AI in Games* 8.3 (2016), pp. 288–300.

[23] Marc Harper, Vince Knight, and Martin Jones. *Axelrod-Python/axelrod-dojo: v0.0.1*. July 2017. DOI: 10.5281/zenodo.824264. URL: https://doi.org/10.5281/zenodo.824264.

[24] Christian Hilbe, Martin A Nowak, and Arne Traulsen. “Adaptive dynamics of extortion and compliance”. In: *PloS one* 8.11 (2013), e77886.

[25] Christian Hilbe et al. “Memory- ¡i¿n¡/i¿ strategies of direct reciprocity”. In: *Proceedings of the National Academy of Sciences* (2017), p. 201621239. ISSN: 0027-8424. DOI: 10.1073/pnas.1621239114. URL: http://www.pnas.org/lookup/doi/10.1073/pnas.1621239114.

[26] John D Hunter. “Matplotlib: A 2D graphics environment”. In: *Computing In Science & Engineering* 9.3 (2007), pp. 90–95.

[27] Graham Kendall, Xin Yao, and Siang Yew Chong. *The iterated prisoners’ dilemma: 20 years on*. Vol. 4. World Scientific, 2007.

[28] Knight, Vincent and Harper, Marc and Glynatsi E., Nikoleta. *Data: Empirical Study of Invasion and Resistance for Iterated Prisoner’s Dilemma Strategies*. May 2017. DOI: ?. URL: ?.

[29] Vincent Knight et al. “An Open Framework for the Reproducible Study of the Iterated Prisoner ’ s Dilemma”. In: (2016).

[30] David Kraines and Vivian Kraines. “Pavlov and the prisoner’s dilemma”. In: *Theory and decision* 26.1 (1989), pp. 47–79.

[31] Steven Kuhn. “Prisoner’s Dilemma”. In: *The Stanford Encyclopedia of Philosophy*. Ed. by Edward N. Zalta. Spring 2017. Metaphysics Research Lab, Stanford University, 2017.

[32] Christopher Lee, Marc Harper, and Dashiell Fryer. “The Art of War: Beyond Memory-one Strategies in Population Games”. In: *Plos One* 10.3 (2015), e0120625. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0120625. URL: http://dx.plos.org/10.1371/journal.pone.0120625.

[33] Jiawei Li and Graham Kendall. “A strategy with novel evolutionary features for the iterated prisoner’s dilemma.” In: *Evolutionary Computation* 17.2 (2009), pp. 257–274. ISSN: 1063-6560. DOI: 10.1162/evco.2009.17.2.257. URL: http://www.ncbi.nlm.nih.gov/pubmed/19413490.

[34] Jiawei Li, Graham Kendall, and Senior Member. "The effect of memory size on the evolutionary stability of strategies in iterated prisoner ' s dilemma". In: X.X (2014), pp. 1–8.

[35] Jiawei Li et al. "Engineering Design of Strategies for Winning Iterated Prisoner ' s Dilemma Competitions". In: 3.4 (2011), pp. 348–360.

[36] LIFL. *PRISON*. 2008. URL: `http://www.lifl.fr/IPD/ipd.frame.html`.

[37] Philippe Mathieu and Jean-Paul Delahaye. "New Winning Strategies for the Iterated Prisoner's Dilemma (Extended Abstract)". In: *14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015)* (2015), pp. 1665–1666. ISSN: 15582914.

[38] Wes McKinney et al. "Data structures for statistical computing in python". In: *Proceedings of the 9th Python in Science Conference*. Vol. 445. van der Voort S, Millman J. 2010, pp. 51–56.

[39] Shashi Mittal and Kalyanmoy Deb. "Optimal strategies of the iterated prisoner's dilemma problem for multiple conflicting objectives". In: *IEEE Transactions on Evolutionary Computation* 13.3 (2009), pp. 554–565. ISSN: 1089778X. DOI: `10.1109/TEVC.2008.2009459`.

[40] P.A.P. Moran. "Random Processes in Genetics". In: April (1957), pp. 60–71.

[41] John H Nachbar. "Evolution in the finitely repeated prisoner's dilemma". In: *Journal of Economic Behavior & Organization* 19.3 (1992), pp. 307–326.

[42] M Nowak and K Sigmund. "A strategy of win-stay, lose-shift that outperforms tit-for-tat in the Prisoner's Dilemma game." In: *Nature* 364.6432 (1993), pp. 56–58. ISSN: 0028-0836. DOI: `10.1038/364056a0`.

[43] Martin A Nowak. *Evolutionary Dynamics: Exploring the Equations of Life.* Cambridge: Harvard University Press. ISBN: 0674023382. DOI: `10.1086/523139`.

[44] William H Press and Freeman J Dyson. "Iterated Prisoner's Dilemma contains strategies that dominate any evolutionary opponent." In: *Proceedings of the National Academy of Sciences of the United States of America* 109.26 (2012), pp. 10409–13. ISSN: 1091-6490. DOI: `10.1073/pnas.1206569109`. URL: `http://www.pnas.org/content/109/26/10409.abstract`.

[45] Andreas Prli and James B. Procter. "Ten Simple Rules for the Open Development of Scientific Software". In: *PLoS Computational Biology* 8.12 (2012), e1002802. ISSN: 1553-7358. DOI: `10.1371/journal.pcbi.1002802`. URL: `http://dx.plos.org/10.1371/journal.pcbi.1002802`.

[46] Arthur Robson. *EFFICIENCY IN EVOLUTIONARY GAMES: DARWIN, NASH AND SECRET HANDSHAKE*. Working Papers. Michigan - Center for Research on Economic & Social Theory, 1989. URL: `http://EconPapers.repec.org/RePEc:fth:michet:89-22`.

[47] Arthur J Robson. "Efficiency in evolutionary games: Darwin, Nash and the secret handshake". In: *Journal of theoretical Biology* 144.3 (1990), pp. 379–396.

[48] Geir Kjetil Sandve et al. "Ten Simple Rules for Reproducible Computational Research". In: *PLoS Computational Biology* 9.10 (2013), pp. 1–4. ISSN: 1553734X. DOI: `10.1371/journal.pcbi.1003285`.

[49] Alexander J. Stewart and Joshua B. Plotkin. "Extortion and cooperation in the Prisoners Dilemma". In: *Proceedings of the National Academy of Sciences* 109.26 (2012), pp. 10134–10135. DOI: `10.1073/pnas.1208087109`. eprint: `http://www.pnas.org/content/109/26/10134.full.pdf`. URL: `http://www.pnas.org/content/109/26/10134.short`.

[50] Alexander J Stewart and Joshua B Plotkin. "From extortion to generosity, evolution in the iterated prisoners dilemma". In: *Proceedings of the National Academy of Sciences* 110.38 (2013), pp. 15348–15353.

[51] The Axelrod project developers. *Axelrod: v2.9.0*. Apr. 2016. DOI: `499122`. URL: `http://dx.doi.org/10.5281/zenodo.499122`.

[52] E Tzafestas. "Toward adaptive cooperative behavior". In: *From Animals to animals: Proceedings of the 6th International Conference on the Simulation of Adaptive Behavior (SAB-2000)* 2 (2000), pp. 334–340.

[53] Unkwown. *www.prisoners-dilemma.com*. 2017. URL: `http://www.prisoners-dilemma.com/`.

[54] Stfan van der Walt, S Chris Colbert, and Gael Varoquaux. "The NumPy array: a structure for efficient numerical computation". In: *Computing in Science & Engineering* 13.2 (2011), pp. 22–30.

[55] Jeffrey West et al. "The prisoners dilemma as a cancer model". In: *Convergent Science Physical Oncology* 2.3 (2016), p. 035002. URL: `http://stacks.iop.org/2057-1739/2/i=3/a=035002`.

[56] Greg Wilson et al. "Best Practices for Scientific Computing". In: 12.1 (2014). DOI: `10.1371/journal.pbio.1001745`.

[57] Jianzhong Wu and Robert Axelrod. "How to cope with noise in the iterated prisoner's dilemma". In: *Journal of Conflict resolution* 39.1 (1995), pp. 183–189.

## A List of players

1. $\phi$ - *Deterministic* - *Memory depth*: $\infty$. [51]
2. $\pi$ - *Deterministic* - *Memory depth*: $\infty$. [51]
3. $e$ - *Deterministic* - *Memory depth*: $\infty$. [51]
4. ALLCorALLD - *Stochastic* - *Memory depth*: 1. [51]
5. Adaptive - *Deterministic* - *Memory depth*: $\infty$. [35]
6. Adaptive Pavlov 2006 - *Deterministic* - *Memory depth*: $\infty$. [27]
7. Adaptive Pavlov 2011 - *Deterministic* - *Memory depth*: $\infty$. [35]
8. Adaptive Tit For Tat: 0.5 - *Deterministic* - *Memory depth*: $\infty$. [52]
9. Aggravater - *Deterministic* - *Memory depth*: $\infty$. [51]
10. Alternator - *Deterministic* - *Memory depth*: 1. [13, 39]
11. Alternator Hunter - *Deterministic* - *Memory depth*: $\infty$. [51]
12. Anti Tit For Tat - *Deterministic* - *Memory depth*: 1. [24]
13. AntiCycler - *Deterministic* - *Memory depth*: $\infty$. [51]
14. Appeaser - *Deterministic* - *Memory depth*: $\infty$. [51]
15. Arrogant QLearner - *Stochastic* - *Memory depth*: $\infty$. [51]
16. Average Copier - *Stochastic* - *Memory depth*: $\infty$. [51]
17. Better and Better - *Stochastic* - *Memory depth*: $\infty$. [36]
18. Bully - *Deterministic* - *Memory depth*: 1. [41]
19. Calculator - *Stochastic* - *Memory depth*: $\infty$. [36]
20. Cautious QLearner - *Stochastic* - *Memory depth*: $\infty$. [51]
21. CollectiveStrategy(**CS**) - *Deterministic* - *Memory depth*: $\infty$. [33]
22. Contrite Tit For Tat(**CTfT**) - *Deterministic* - *Memory depth*: 3. [57]
23. Cooperator - *Deterministic* - *Memory depth*: 0. [13, 39, 44]
24. Cooperator Hunter - *Deterministic* - *Memory depth*: $\infty$. [51]
25. Cycle Hunter - *Deterministic* - *Memory depth*: $\infty$. [51]
26. Cycler CCCCCD - *Deterministic* - *Memory depth*: 5. [51]
27. Cycler CCCD - *Deterministic* - *Memory depth*: 3. [51]
28. Cycler CCCDCD - *Deterministic* - *Memory depth*: 5. [51]
29. Cycler CCD - *Deterministic* - *Memory depth*: 2. [39]
30. Cycler DC - *Deterministic* - *Memory depth*: 1. [51]
31. Cycler DDC - *Deterministic* - *Memory depth*: 2. [39]
32. Davis: 10 - *Deterministic* - *Memory depth*: $\infty$. [12]
33. Defector - *Deterministic* - *Memory depth*: 0. [13, 39, 44]
34. Defector Hunter - *Deterministic* - *Memory depth*: $\infty$. [51]
35. Desperate - *Stochastic* - *Memory depth*: 1. [17]
36. Doubler - *Deterministic* - *Memory depth*: $\infty$. [36]
37. EasyGo - *Deterministic* - *Memory depth*: $\infty$. [35, 36]
38. Eatherley - *Stochastic* - *Memory depth*: $\infty$. [11]
39. Eventual Cycle Hunter - *Deterministic* - *Memory depth*: $\infty$. [51]
40. Evolved ANN - *Deterministic* - *Memory depth*: $\infty$. [51]
41. Evolved ANN 5 - *Deterministic* - *Memory depth*: $\infty$. [51]
42. Evolved ANN 5 Noise 05 - *Deterministic* - *Memory depth*: $\infty$. [51]
43. Evolved FSM 16 - *Deterministic* - *Memory depth*: 16. [51]
44. Evolved FSM 16 Noise 05 - *Deterministic* - *Memory depth*: 16. [51]
45. Evolved FSM 4 - *Deterministic* - *Memory depth*: 4. [51]
46. Evolved HMM 5 - *Stochastic* - *Memory depth*: 5. [51]
47. EvolvedLookerUp1_1_1 - *Deterministic* - *Memory depth*: $\infty$. [51]
48. EvolvedLookerUp2_2_2 - *Deterministic* - *Memory depth*: $\infty$. [51]
49. FSM Player: [(0, 'C', 0, 'C'), (0, 'D', 3, 'C'), (1, 'C', 5, 'D'), (1, 'D', 0, 'C'), (2, 'C', 3, 'C'), (2, 'D', 2, 'D'), (3, 'C', 4, 'D'), (3, 'D', 6, 'D'), (4, 'C', 3, 'C'), (4, 'D', 1, 'D'), (5, 'C', 6, 'C'), (5, 'D', 3, 'D'), (6, 'C', 6, 'D'), (6, 'D', 6, 'D'), (7, 'C', 7, 'D'), (7, 'D', 5, 'C')], 0, C(**TF3**) - *Deterministic* - *Memory depth*: $\infty$.

50. FSM Player: [(0, 'C', 13, 'D'), (0, 'D', 12, 'D'), (1, 'C', 3, 'D'), (1, 'D', 4, 'D'), (2, 'C', 14, 'D'), (2, 'D', 9, 'D'), (3, 'C', 0, 'C'), (3, 'D', 1, 'D'), (4, 'C', 1, 'D'), (4, 'D', 2, 'D'), (5, 'C', 12, 'C'), (5, 'D', 6, 'C'), (6, 'C', 1, 'C'), (6, 'D', 14, 'D'), (7, 'C', 12, 'D'), (7, 'D', 2, 'D'), (8, 'C', 7, 'D'), (8, 'D', 9, 'D'), (9, 'C', 8, 'D'), (9, 'D', 0, 'D'), (10, 'C', 2, 'C'), (10, 'D', 15, 'C'), (11, 'C', 7, 'D'), (11, 'D', 13, 'D'), (12, 'C', 3, 'C'), (12, 'D', 8, 'D'), (13, 'C', 7, 'C'), (13, 'D', 10, 'D'), (14, 'C', 10, 'D'), (14, 'D', 7, 'D'), (15, 'C', 15, 'C'), (15, 'D', 11, 'D')], 0, C(**TF2**) - *Deterministic* - *Memory depth*: ∞.

51. FSM Player: [(0, 'C', 7, 'C'), (0, 'D', 1, 'C'), (1, 'C', 11, 'D'), (1, 'D', 11, 'D'), (2, 'C', 8, 'D'), (2, 'D', 8, 'C'), (3, 'C', 3, 'C'), (3, 'D', 12, 'D'), (4, 'C', 6, 'C'), (4, 'D', 3, 'C'), (5, 'C', 11, 'C'), (5, 'D', 8, 'D'), (6, 'C', 13, 'D'), (6, 'D', 14, 'C'), (7, 'C', 4, 'D'), (7, 'D', 2, 'D'), (8, 'C', 14, 'D'), (8, 'D', 8, 'D'), (9, 'C', 0, 'C'), (9, 'D', 10, 'D'), (10, 'C', 8, 'C'), (10, 'D', 15, 'C'), (11, 'C', 6, 'D'), (11, 'D', 5, 'D'), (12, 'C', 6, 'D'), (12, 'D', 9, 'D'), (13, 'C', 9, 'D'), (13, 'D', 8, 'D'), (14, 'C', 8, 'D'), (14, 'D', 13, 'D'), (15, 'C', 4, 'C'), (15, 'D', 5, 'C')], 0, C(**TF1**) - *Deterministic* - *Memory depth*: ∞.

52. Feld: 1.0, 0.5, 200 - *Stochastic* - *Memory depth*: 200. [12]

53. Firm But Fair - *Stochastic* - *Memory depth*: 1. [21]

54. Fool Me Forever - *Deterministic* - *Memory depth*: ∞. [51]

55. Fool Me Once - *Deterministic* - *Memory depth*: ∞. [51]

56. Forgetful Fool Me Once: 0.05 - *Stochastic* - *Memory depth*: ∞. [51]

57. Forgetful Grudger - *Deterministic* - *Memory depth*: 10. [51]

58. Forgiver - *Deterministic* - *Memory depth*: ∞. [51]

59. Forgiving Tit For Tat(**FTfT**) - *Deterministic* - *Memory depth*: ∞. [51]

60. Fortress3 - *Deterministic* - *Memory depth*: 3. [8]

61. Fortress4 - *Deterministic* - *Memory depth*: 4. [8]

62. GTFT: 0.33 - *Stochastic* - *Memory depth*: 1. [22, 42]

63. General Soft Grudger: n=1,d=4,c=2 - *Deterministic* - *Memory depth*: ∞. [51]

64. Gradual - *Deterministic* - *Memory depth*: ∞. [16]

65. Gradual Killer: ('D', 'D', 'D', 'D', 'D', 'C', 'C') - *Deterministic* - *Memory depth*: ∞. [36]

66. Grofman - *Stochastic* - *Memory depth*: ∞. [12]

67. Grudger - *Deterministic* - *Memory depth*: ∞. [12, 15, 16, 17, 35]

68. GrudgerAlternator - *Deterministic* - *Memory depth*: ∞. [36]

69. Grumpy: Nice, 10, -10 - *Deterministic* - *Memory depth*: ∞. [51]

70. Handshake - *Deterministic* - *Memory depth*: ∞. [47]

71. Hard Go By Majority - *Deterministic* - *Memory depth*: ∞. [39]

72. Hard Go By Majority: 10 - *Deterministic* - *Memory depth*: 10. [51]

73. Hard Go By Majority: 20 - *Deterministic* - *Memory depth*: 20. [51]

74. Hard Go By Majority: 40 - *Deterministic* - *Memory depth*: 40. [51]

75. Hard Go By Majority: 5 - *Deterministic* - *Memory depth*: 5. [51]

76. Hard Prober - *Deterministic* - *Memory depth*: ∞. [36]

77. Hard Tit For 2 Tats(**HTf2T**) - *Deterministic* - *Memory depth*: 3. [49]

78. Hard Tit For Tat(**HTfT**) - *Deterministic* - *Memory depth*: 3. [53]

79. Hesitant QLearner - *Stochastic* - *Memory depth*: ∞. [51]

80. Hopeless - *Stochastic* - *Memory depth*: 1. [17]

81. Inverse - *Stochastic* - *Memory depth*: ∞. [51]

82. Inverse Punisher - *Deterministic* - *Memory depth*: ∞. [51]

83. Joss: 0.9 - *Stochastic* - *Memory depth*: 1. [12, 49]

84. Level Punisher - *Deterministic* - *Memory depth*: ∞. [19]

85. Limited Retaliate 2: 0.08, 15 - *Deterministic* - *Memory depth*: ∞. [51]

86. Limited Retaliate 3: 0.05, 20 - *Deterministic* - *Memory depth*: ∞. [51]

87. Limited Retaliate: 0.1, 20 - *Deterministic* - *Memory depth*: ∞. [51]

88. MEM2 - *Deterministic* - *Memory depth*: ∞. [34]

89. Math Constant Hunter - *Deterministic* - *Memory depth*: ∞. [51]

90. Meta Hunter Aggressive: 7 players - *Deterministic* - *Memory depth*: ∞. [51]

91. Meta Hunter: 6 players - *Deterministic* - *Memory depth*: ∞. [51]

92. Naive Prober: 0.1 - *Stochastic* - *Memory depth*: 1. [35]
93. Negation - *Stochastic* - *Memory depth*: 1. [53]
94. Nice Average Copier - *Stochastic* - *Memory depth*: $\infty$. [51]
95. Nydegger - *Deterministic* - *Memory depth*: 3. [12]
96. Omega TFT: 3, 8 - *Deterministic* - *Memory depth*: $\infty$. [27]
97. Once Bitten - *Deterministic* - *Memory depth*: 12. [51]
98. Opposite Grudger - *Deterministic* - *Memory depth*: $\infty$. [51]
99. PSO Gambler 1_1_1 - *Stochastic* - *Memory depth*: $\infty$. [51]
100. PSO Gambler 2_2_2 - *Stochastic* - *Memory depth*: $\infty$. [51]
101. PSO Gambler 2_2_2 Noise 05 - *Stochastic* - *Memory depth*: $\infty$. [51]
102. PSO Gambler Mem1 - *Stochastic* - *Memory depth*: 1. [51]
103. Predator - *Deterministic* - *Memory depth*: 9. [8]
104. Prober - *Deterministic* - *Memory depth*: $\infty$. [35]
105. Prober 2 - *Deterministic* - *Memory depth*: $\infty$. [36]
106. Prober 3 - *Deterministic* - *Memory depth*: $\infty$. [36]
107. Prober 4 - *Deterministic* - *Memory depth*: $\infty$. [36]
108. Pun1 - *Deterministic* - *Memory depth*: 2. [7]
109. Punisher - *Deterministic* - *Memory depth*: $\infty$. [51]
110. Raider - *Deterministic* - *Memory depth*: 3. [9]
111. Random Hunter - *Deterministic* - *Memory depth*: $\infty$. [51]
112. Random: 0.5 - *Stochastic* - *Memory depth*: 0. [12, 52]
113. Remorseful Prober: 0.1 - *Stochastic* - *Memory depth*: 2. [35]
114. Resurrection - *Deterministic* - *Memory depth*: 1. [19]
115. Retaliate 2: 0.08 - *Deterministic* - *Memory depth*: $\infty$. [51]
116. Retaliate 3: 0.05 - *Deterministic* - *Memory depth*: $\infty$. [51]
117. Retaliate: 0.1 - *Deterministic* - *Memory depth*: $\infty$. [51]
118. Revised Downing: True - *Deterministic* - *Memory depth*: $\infty$. [12]
119. Ripoff - *Deterministic* - *Memory depth*: 2. [6]
120. Risky QLearner - *Stochastic* - *Memory depth*: $\infty$. [51]
121. SelfSteem - *Stochastic* - *Memory depth*: $\infty$. [18]
122. ShortMem - *Deterministic* - *Memory depth*: 10. [18]
123. Shubik - *Deterministic* - *Memory depth*: $\infty$. [12]
124. Slow Tit For Two Tats - *Deterministic* - *Memory depth*: 2. [51]
125. Slow Tit For Two Tats 2 - *Deterministic* - *Memory depth*: 2. [36]
126. Sneaky Tit For Tat - *Deterministic* - *Memory depth*: $\infty$. [51]
127. Soft Go By Majority - *Deterministic* - *Memory depth*: $\infty$. [13, 39]
128. Soft Go By Majority: 10 - *Deterministic* - *Memory depth*: 10. [51]
129. Soft Go By Majority: 20 - *Deterministic* - *Memory depth*: 20. [51]
130. Soft Go By Majority: 40 - *Deterministic* - *Memory depth*: 40. [51]
131. Soft Go By Majority: 5 - *Deterministic* - *Memory depth*: 5. [51]
132. Soft Grudger - *Deterministic* - *Memory depth*: 6. [35]
133. Soft Joss: 0.9 - *Stochastic* - *Memory depth*: 1. [36]
134. SolutionB1 - *Deterministic* - *Memory depth*: 3. [5]
135. SolutionB5 - *Deterministic* - *Memory depth*: 5. [5]
136. Spiteful Tit For Tat - *Deterministic* - *Memory depth*: $\infty$. [36]
137. Stochastic Cooperator - *Stochastic* - *Memory depth*: 1. [1]
138. Stochastic WSLS: 0.05 - *Stochastic* - *Memory depth*: 1. [51]
139. Suspicious Tit For Tat - *Deterministic* - *Memory depth*: 1. [16, 24]
140. Tester - *Deterministic* - *Memory depth*: $\infty$. [11]
141. ThueMorse - *Deterministic* - *Memory depth*: $\infty$. [51]
142. ThueMorseInverse - *Deterministic* - *Memory depth*: $\infty$. [51]
143. Thumper - *Deterministic* - *Memory depth*: 2. [6]
144. Tit For 2 Tats(**Tf2T**) - *Deterministic* - *Memory depth*: 2. [13]

145. Tit For Tat(**TfT**) - *Deterministic* - *Memory depth*: 1. [12]

146. Tricky Cooperator - *Deterministic* - *Memory depth*: 10. [51]

147. Tricky Defector - *Deterministic* - *Memory depth*: $\infty$. [51]

148. Tullock: 11 - *Stochastic* - *Memory depth*: 11. [12]

149. Two Tits For Tat(**2TfT**) - *Deterministic* - *Memory depth*: 2. [13]

150. VeryBad - *Deterministic* - *Memory depth*: $\infty$. [18]

151. Willing - *Stochastic* - *Memory depth*: 1. [17]

152. Win-Shift Lose-Stay: D(**WShLSt**) - *Deterministic* - *Memory depth*: 1. [35]

153. Win-Stay Lose-Shift: C(**WSLS**) - *Deterministic* - *Memory depth*: 1. [30, 42, 49]

154. Winner12 - *Deterministic* - *Memory depth*: 2. [37]

155. Winner21 - *Deterministic* - *Memory depth*: 2. [37]

156. Worse and Worse - *Stochastic* - *Memory depth*: $\infty$. [36]

157. Worse and Worse 2 - *Stochastic* - *Memory depth*: $\infty$. [36]

158. Worse and Worse 3 - *Stochastic* - *Memory depth*: $\infty$. [36]

159. ZD-Extort-2 v2: 0.125, 0.5, 1 - *Stochastic* - *Memory depth*: 1. [31]

160. ZD-Extort-2: 0.1111111111111111, 0.5 - *Stochastic* - *Memory depth*: 1. [49]

161. ZD-Extort-4: 0.23529411764705882, 0.25, 1 - *Stochastic* - *Memory depth*: 1. [51]

162. ZD-GEN-2: 0.125, 0.5, 3 - *Stochastic* - *Memory depth*: 1. [31]

163. ZD-GTFT-2: 0.25, 0.5 - *Stochastic* - *Memory depth*: 1. [49]

164. ZD-SET-2: 0.25, 0.0, 2 - *Stochastic* - *Memory depth*: 1. [31]