A Round-Robin Tournament of the Iterated Prisoner's Dilemma with Complete Memory-Size-Three Strategies

Tobias Kretz

PTV – Planung Transport Verkehr AG
Stumpfstraße 1
D-76131 Karlsruhe, Germany
Tobias.Kretz@ptv.de

The results of simulating a prisoner's dilemma round-robin tournament are presented. In the tournament, each participating strategy played an iterated prisoner's dilemma against each of the other strategies (round-robin) and as a variant also against itself. The participants of a tournament are all deterministic strategies and have the same memory size regarding their own and their opponent's past actions. Memory sizes of up to three of the most recent actions of their opponent and up to two of their own are discussed. The investigation focused on the influence of the number of iterations, the details of the payoff matrix, and the memory size. The main result for the tournament as carried out here is that different strategies emerge as winners for different payoff matrices. This is true even for different payoff matrices that are judged to be similar if they fulfill relations $T + S = P + R$ or $2R > T + S$. As a consequence of this result, it is suggested that whenever the iterated prisoner's dilemma is used to model a real system that does not explicitly fix the payoff matrix, conclusions should be checked for validity when a different payoff matrix is used.

1. Introduction and Motivation

The prisoner's dilemma [1, 2] is probably the most prominent and discussed example from game theory, which is a result of its standing as the model of the formation of cooperation in the course of biological as well as cultural evolution [2, 3].

A naive interpretation of Darwin's theory might suggest that evolution favors nothing but direct battle and plain competition. However, numerous observations of cooperation in the animal kingdom oppose this idea by plain evidence. While such examples among animals are impressive, clearly the most complex and complicated interplay of cooperation and competition occurs with humans, a fact that becomes obvious when a large number of humans gather as a crowd in spatial proximity. There are astonishing and well-known examples for both: altruism among strangers under dangerous external conditions [4–11] as well as fierce competition for goods with very limited material value often linked with a lack of information [12, 13]. For examples of behavior between these two extremes, see the overviews in [14, 15]. In relation to these events, and possible similar future events of pedestrian and evacuation dynamics [16], the widespread naive interpretation of the theory of evolution in a sense poses a danger. It might give people in such situations the wrong idea of what their surrounding fellows are going to do and suggest overly competitive or dangerous behavior. Knowledge of certain historic events, together with theories that suggest why cooperation against immediate maximal self-benefit can be rational, hopefully can immunize against such destructive thoughts and actions.

From the beginning, the prisoner's dilemma was investigated in an iterated way [17, 18]. Often included was the ability of strategies to hark back on the course of tournament events [2, 19] without limit, that is, their memory potentially included every one of their own and their opponents' steps. Despite the possibility of using more memory, the first strategy to emerge as a winner, tit-for-tat (TFT), got along with a memory of only the most recent action of their opponent. Another famous and successful strategy, Pavlov, also uses a small memory: it just needs to remember its own and the opponent's action. In this paper the effect of extending memory up to the three latest actions of the opponent and up to their own two latest actions is investigated.

In the course of discussing the prisoner's dilemma a number of methods have been introduced such as probabilistic strategies to model errors ("noise") [20], evolutionary (ecologic) investigation [2], spatial relations (players only play against neighboring opponents) [21–30], and creating strategies by genetic programming [3, 20, 31–33]. Most of these can be combined. For an overview on further variants, see [34, 35].

Contrary to more elaborate methods, a main focus in this work is to avoid arbitrary and probabilistic decisions such as choosing a subset of strategies of a class, or to locate strategies spatially in neighborhoods. Such spatial variants, as well as genetic approaches, are excluded. Instead, each strategy of the class participates and plays against each other. A consequence of investigating complete classes and avoiding arbitrariness is that using probabilistic strategies is difficult. In general, infinitely many rules could be constructed from a memory state with the infinitely many real numbers that can serve as values for the probability. Selecting some of the numbers to be used and rejecting others would have to be based on elementary reasoning to avoid arbitrariness. It can be argued that there are elementary ways to calculate the probability for cooperation, for example, a linear function of the ratio of cooperation of the opponent. Nevertheless, while some ways to calculate the probability are more elementary than others, it is not clear which calculations are still elementary and which are not. Therefore, in this contribution no probabilistic strategies are considered.

The round-robin model as well—at least in parts—is a consequence of avoiding arbitrariness. For example, drawing lots to choose pairs of competitors as in a tournament would bring in a probabilistic element. In other words: the source code written for this investigation does not at any point make use of random numbers. It is a deterministic brute force calculation of a large number of strategies and a very large number of single games. The relevance lies not in modeling a specific system of reality, but in the completeness of the investigated class and in general the small degree of freedom (arbitrariness) of the system.

By the strictness and generality of the procedure, a strategy can be seen as a Mealy automaton or the iterative game between two strategies as a Moore machine [36–39]; respectively, a spatially zero-dimensional cellular automaton [40, 41] (see Section 3).

2. Definition of a Strategy

In the sense of this paper a strategy with a memory size n has $n+1$ substrategies to define the action in the first, second, ... n^{th}, and any further iteration. The substrategy for the first iteration decides how to start the tournament, the substrategy for the second iteration depends on the action(s) of the first iteration, the substrategy for the third iteration depends on the actions in the first and second iterations (if memory size is larger than one), and the substrategy for the $(N > n)^{\text{th}}$ iteration depends on the actions in the $(N-n)$ to $(N-1)^{\text{st}}$ iterations (compare Figure 1).

A similar approach was followed in [42], but there are differences in the definition of the class concerning behavior in the first $n-1$ iterations. Most importantly their approach did not use a round-robin tournament with all strategies of a class, but was combined with a genetic approach.

Another investigation dealing with the effects of memory size is [43], but their strategies were probabilistic and therefore not all strategies participated in the process.

2.1 Data Size of a Strategy, Number of Strategies, and Number of Games

In the first round of an iterated game there is no information from the opponent, so the strategy consists of deciding how to begin (one bit). In the second round, there is only information on one past step from the opponent, so the strategy includes deciding how to react (two bits). The third round is still part of the starting phase and therefore also has its own part of the strategy (four bits, if the decision does not depend on a strategy's own preceding action). Therefore, there are 128 strategies when using a no-own-two-opponent memory. Finally, there are eight more bits with size-three memory. An example of calculating the number combination (1/2/12/240) from the TFT strategy is shown in Figure 1. These 15 bits lead to a total of $N = 32\,768$ different strategies. If each strategy plays against every other strategy and against itself, there are $N \cdot (N+1) / 2 = 2^{29}$ different iterated prisoner's dilemmas to calculate.

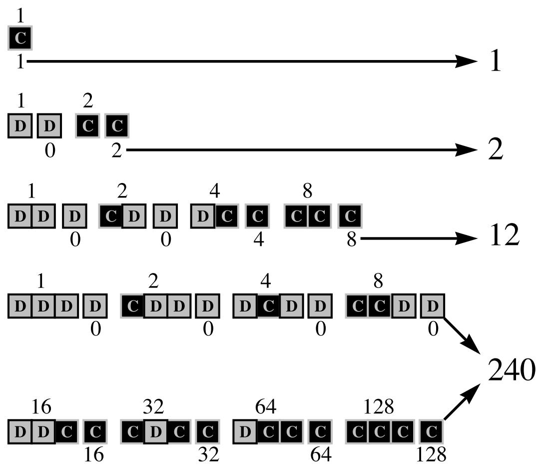

Figure 1. TFT as strategy (1/2/12/240). The part (1/2/12) applies during the starting phase when only zero, one, or two earlier states of the opponent exist. Cooperation is coded with a "1", defection with a "0". If a strategy also remembers its own past actions the information is always stored in the lower bits. For example, with the triples the leftmost would indicate a strategy's own preceding action and the middle and right would indicate the second-to-last and last action of the opponent ("low to high" is "left to right").

Table 1 summarizes these numbers for different memory sizes. To remember the last n actions of a pair of strategies, $2\,n$ bits are needed. For the results of a strategy over the entire course of iterations a few bytes are needed for each pair of strategies, depending on the kind of evaluation. The number of pairs of strategies—and this is the limiting component—grows at least approximately like $2^{2^{n+2}-3}$. On today's common PCs RAM demands are therefore trivial up to a memory size of $n = 2$, in the lower range of 64-bit technology (some GBs of RAM) for $n = 3$, and totally unavailable for $n = 4$ and larger (more than an exabyte).

Memory Size self /other	#Bits	#Strategies	#Games in One Iteration
0 / 0	1	2	1 resp. 3
0 / 1	3	8	28 resp. 36
1 / 1	5	32	496 resp. 528
0 / 2	7	128	8128 resp. 8256
1 / 2	13	8192	$\approx 33.55 \cdot 10^6$
2 / 1	13	8192	$\approx 33.55 \cdot 10^6$
0 / 3	15	32 768	$\approx 536.8 \cdot 10^6$
2 / 2	21	2 097 152	$\approx 2.199 \cdot 10^9$
1 / 3	29	536 870 912	$\approx 144.1 \cdot 10^{15}$
3 / 1	29	536 870 912	$\approx 144.1 \cdot 10^{15}$
0 / 4	31	2 147 483 648	$\approx 2.306 \cdot 10^{18}$

Table 1. Number of bits (b) to represent a strategy, number of strategies (2^b), and number of prisoner's dilemma games in an iteration step in a round-robin tournament $\left(2^{b-1}\left(2^b \pm 1\right)\right)$ for different memory sizes. This leads to the computational effort shown in Table 2.

Memory Size self /other	RAM	Time
0 / 0	10 B	insignificant
0 / 1	100 B	insignificant
1 / 1	10 KB	s .. min
0 / 2	100 KB	s .. min
1 / 2	100 MB	min .. d
2 / 1	100 MB	min .. d
0 / 3	10 GB	h .. weeks
2 / 2	10 TB	d .. year
1 / 3	1 EB	> year
3 / 1	1 EB	> year
0 / 4	10 EB	decade(s) (?)

Table 2. Magnitudes of computational resource requirements (on a double quad core Intel Xeon 5320). The computation time depends significantly on the number of different payoff matrices being investigated. Large scale simulations with parallel computing of the iterated prisoner's dilemma has also been dealt with in [44].

3. The Cellular Automata Perspective

This section presents the system in terms of cellular automata. This can help obtain a visual depiction of the system dynamics. However, the reader may well skip this and proceed to Section 4.

Wolfram's elementary cellular automata are defined (or interpreted) to exist in one spatial plus one temporal dimension. However, the rules can also be applied to a point-like cellular automaton with memory as shown in Figure 2. This system can be interpreted as a cellular automaton that has a memory and a binary state, or as an automaton that can have one of eight states with restricted transitions between the states. For the full set of 256 rules each state can be reached in principle from two other states. Also, from a particular state two states can be reached. Choosing a specific rule is selecting one incoming and one outgoing state. This is exemplified in Figure 3 for rule 110. For the iterated prisoner's dilemma two such cellular automata need to interact and determine their next state from the data of the other automaton as shown in Figure 4. It is of course possible to interpret two interacting cellular automata as one single point-like cellular automaton with a larger set of states. Then, Figure 4 would translate into Figure 5. A transition graph could be drawn (with 64 nodes that all have one of four possible incoming and outgoing links or a specific combination of rules) for further theoretical analysis. For now we abandon these basic and theoretical considerations and just adhere to the fact that the implementation of the process can be seen as a cellular automaton. Or, more precisely, as an enormous number of combinations of interacting very simple cellular automata.

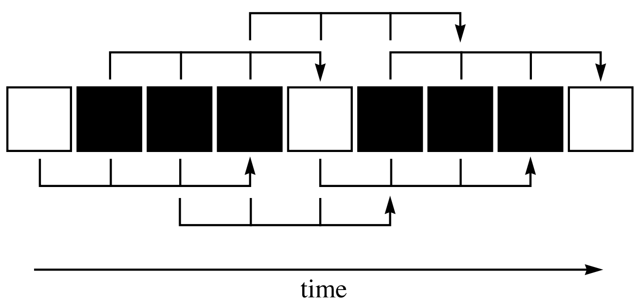

Figure 2. Rule 110 applied self-referentially to a point-like cellular automaton with memory. Note: as time increases toward the right and the most recent state is meant to be stored in the highest bit, but higher bits are written to the left, we have to reverse the bits compared to Wolfram's standard notation.

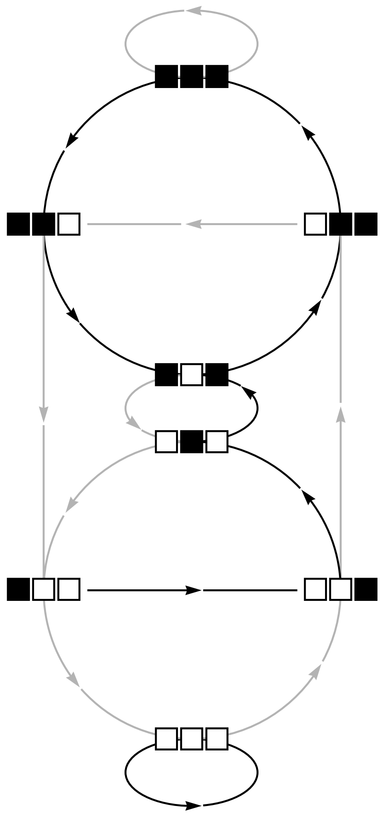

Figure 3. Transition graph for rule 110 (black links) and possible links or other rules (gray links).

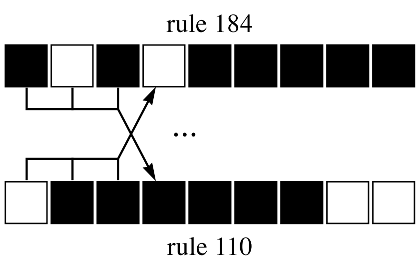

Figure 4. Rules 184 and 110 interacting. For the iterated prisoner's dilemma, the dependence here models the situation when a prisoner remembers the three preceding moves of the opponent but none of its own.

Figure 5. Figure 4 depicted as a single cellular automaton. If the states of both automata are white (black) the state here is shown as well as white (black). If 184 is white (black) and 110 black (white), the state here is yellow (red).

4. Payoff Matrix

The four values T, R, P, and S of the payoff matrix (see Table 3) need to fulfill the relation

$$T > R > P > S \tag{1}$$

to be faced with a prisoner's dilemma. For the purpose of this contribution $S = 0$ can be chosen without loss of generality, as whenever the payoff matrix is applied all strategies have played the same number of games. In addition to equation (1) it is often postulated that

$$2R > T \tag{2}$$

holds.

	C(2)	D(2)
C(1)	$R\ R$	$S\ T$
D(1)	$T\ S$	$P\ P$

Table 3. General payoff matrix.

The equation

$$T + S = P + R \tag{3}$$

marks a special set of payoff matrices with values that can be seen as a model of a trading process. Here, the exchanged good has a higher value for the buyer i than the seller j:

$$p_{ij} = \alpha + \beta\delta_i - \gamma\delta_j \tag{4}$$

with $\delta = 1$ if a player cooperates and $\delta = 0$ for defection. Therefore, β can be interpreted as the "gain from receiving" value and γ the "cost of giving" value. α is a constant to guarantee $p_{ij} \geq 0$. For technical convenience, T, R, P, and S can be calculated from these: $T = \alpha + \beta$, $R = \alpha + \beta - \gamma$, $P = \alpha$, and $S = \alpha - \gamma$. Aside from the descriptive interpretation as gain from receiving and cost of giving, this reparametrization has the advantage that the original condition equation (1) and the additional conditions equation (2) and $S = 0$ reduce to $\beta > \gamma = \alpha$. Furthermore, it is the form of the basic equation in G. Price's model for the evolution of cooperation [45, 46].

Because we want to investigate more than payoff matrices, where equations (2) and (3) hold, we rewrite

$$T = (1 + a + b)\,P, \tag{5}$$

$$R = (1 + a)\,P, \tag{6}$$

$$a = \frac{\alpha}{P} - 1 > 0, \tag{7}$$

$$b = \frac{\beta}{P} > 0. \tag{8}$$

In principle, we could set $P = 1$ without loss of generality. However, we cannot set $P = 1$ while requiring that T and R are integers and that all combinations of hold or do not hold in equations (2) and (3) are generated.

Now, equation (3) can be written as

$$b = 1 \tag{9}$$

and investigated as one variant next to $b > 1$ and $b < 1$. And equation (2) can be written as

$$a + 1 > b. \tag{10}$$

$a + 1 = b$ and $a + 1 < b$ will also be investigated (always taking care that $a > 0$ and $b > 0$ hold). Finally, $a\,(<,=,>)\,1$ and $a\,(<,=,>)\,b$ are relevant conditions, if it is possible to distinguish in this way.

Obviously, not all combinations of these conditions can hold simultaneously. For example, $(a + 1 < b,\ b < 1)$ has no allowed solution. The allowed combinations and the values for T, R, and P are shown in Table 4. For each combination of conditions an infinite number of values could have been found. One could have chosen to interpret ">" as "much greater than" but then selecting specific numbers in a way would have been arbitrary. So the smallest numbers to fulfill a set of conditions have been chosen as representatives.

Cond. 1	Cond. 2	Cond. 3	T	R	P	$T = R + P$	$2R > T$
$b = 1$	$a = 1$		3	2	1	holds	holds
$b = 1$	$a > 1$		4	3	1	holds	holds
$b = 1$	$a < 1$		5	3	2	holds	holds
$b < 1$	$a = 1$		5	4	2		holds
$b < 1$	$a > 1$		6	5	2		holds
$b < 1$	$a < 1$	$b = a$	4	3	2		holds
$b < 1$	$a < 1$	$b > a$	6	4	3		holds
$b < 1$	$a < 1$	$b < a$	6	5	3		holds
$b > 1$	$b < a + 1$	$a > 1$	5	3	1		holds
$b > 1$	$b < a + 1$	$a = 1$	7	4	2		holds
$b > 1$	$b < a + 1$	$a < 1$	9	5	3		holds
$b > 1$	$b = a + 1$	$a = 1$	4	2	1		
$b > 1$	$b > a + 1$	$a = 1$	5	2	1		
$b > 1$	$b = a + 1$	$a > 1$	6	3	1		
$b > 1$	$b > a + 1$	$a > 1$	7	3	1		
$b > 1$	$b = a + 1$	$a < 1$	6	3	2		
$b > 1$	$b > a + 1$	$a < 1$	7	3	2		

Table 4. Investigated variants of payoff matrix values.

5. Iteration, Tournament, and Scoring

In an iteration step all strategies play a prisoner's dilemma against any of the other strategies and themselves. A strategy calculates its action from the preceding actions of the specific opponent. How often strategy i received a T, R, P, or S payoff playing against a specific strategy j is tracked by the counters $N^t_{i\,j}, N^r_{i\,j}, N^p_{i\,j}, N^s_{i\,j}$, that is, in each iteration step for each i and each j one of the four $N^x_{i\,j}$ is increased by 1.

Now, all the payoff matrices from Table 4 are applied one after the other to calculate the total payoff G_i^1 for each payoff matrix and each strategy i:

$$G_i^1 = \sum_j T\, N^T_{i\,j} + R\, N^R_{i\,j} + P\, N^P_{i\,j}. \tag{11}$$

The strategy (or set of strategies) i to yield the highest G_i^1 is one of the main results for a specific iteration round and a specific payoff matrix.

Then, the tournament begins. Each tournament round g is started by calculating the average payoff of the preceding tournament round:

$$G^g = \frac{\sum_i G_i^g\, \delta_i^g}{\sum_i \delta_i^g} \tag{12}$$

where $\delta_i^g = 1$, if strategy i was still participating in round g and $\delta_i^g = 0$ otherwise. Then, δ_i^{g+1} is set to 0, if $\delta_i^g = 0$, or if a strategy scored below average:

$$G_i^g < G^g. \tag{13}$$

The payoff for the next tournament round $g + 1$ is then calculated for all strategies still participating:

$$G_i^{g+1} = \sum_j \left(T\, N_{ij}^T + R\, N_{ij}^R + P\, N_{ij}^P\right) \delta_j^{g+1}. \tag{14}$$

The tournament ends if only one strategy remains or if all remaining strategies score equally in a round (i.e., they have identical G_i^g). The strategies that manage to emerge as winners of such a tournament are the second main result for a specific iteration step and a specific payoff matrix.

Such an elimination tournament can be interpreted as an evolutionary tournament, where the frequency values for the strategies can only take the values $f = 0$ and $f = 1$.

To state it explicitly: all strategies participate in the next iteration step for another first round of the tournament. The elimination process takes place within a step and not across iteration steps. No prisoner's dilemma game is played during or between the rounds of a tournament. Because all strategies are deterministic, this procedure is equivalent to playing the prisoner's dilemma a fixed number of iterations, evaluating the scores, eliminating all strategies that score below average, and again playing a fixed number of iterations with the remaining strategies, and so on.

6. Results

In this section we investigate all payoff matrices listed in Table 4. The strategies are given that have the highest payoff G_i^1 in the first round of the tournament for large numbers of iterations. The winning strategy is given if the system stabilizes to one winner. Additionally, the iteration round that the winning strategy first appears in is given. This implies that for a certain payoff matrix the number of iterations prior to finding the winning strategy is important for determining which strategy will emerge as the best (in the sense described in Section 5).

6.1 Results for No-Own-One-Opponent Memory

With only one action to remember, there are just eight strategies (named (0/0) to (1/3) where (0/0) never cooperates and (1/3) always cooperates). The strategy TFT is (1/2). The simulation ran for 1000 iterations. It is safe to say that this is sufficiently long, as the results—shown in Tables 5 and 6—stabilize at the latest in iteration 16 (respectively 179).

T	R	P	First It.	G_i^1	Tournament
3	2	1	8	(0/0)	(1/2)
4	3	1	4	(0/0)	(1/2)
5	3	2	16	(0/0)	(1/2)
5	4	2	6	(0/0)	(1/2)
6	5	2	4	(0/0)	(1/2)
4	3	2	10	(0/0)	(1/2)
6	4	3	14	(0/0)	(1/2)
6	5	3	6	(0/0)	(1/2)
5	3	1	4	(0/0)	(0/0)
7	4	2	4	(0/0)	(0/0)
9	5	3	4	(0/0)	(0/0)
4	2	1	4	(0/0)	(0/0)
5	2	1	4	(0/0)	(0/0)
6	3	1	4	(0/0)	(0/0)
7	3	1	4	(0/0)	(0/0)
6	3	2	4	(0/0)	(0/0)
7	3	2	4	(0/0)	(0/0)

Table 5. Results for (no own / one opponent) memory, if strategies also play against themselves. "First It." denotes the iteration round after which the results remain the same until iteration 1000. TFT wins the tournament if $b \leq 1$ (regardless of a), while a comparison of the whole set of strategies is won by defect always (ALLD).

6.2 Results for One-Own-One-Opponent Memory

Beginning with the second iteration step under this configuration, strategies base their decision on two bits; one (the higher bit) encodes the previous action of their opponent and the other remembers their own action. Table 7 gives an overview of strategy numbers and compares their behavior.

For this and all further settings 10 000 iterations (and in special cases more) have been simulated. Results are shown in Tables 8 and 9.

T	R	P	First It.	G_i^1	Tournament
3	2	1	8	(0/0)	(0/0)
4	3	1	8	(0/0)	(0/0)
5	3	2	12	(0/0)	(0/0)
5	4	2	162 (2)	(0/0)	(0/2), *(1/2)*
6	5	2	179 (2)	(0/0)	(0/2), *(1/2)*
4	3	2	108 (2)	(0/0)	(0/0), *(0/2)*
6	4	3	168 (2)	(0/0)	(0/0), *(0/2)*
6	5	3	80 (2)	(0/0)	(0/0), *(0/2)*
5	3	1	4	(0/0)	(0/0)
7	4	2	7	(0/0)	(0/0)
9	5	3	8	(0/0)	(0/0)
4	2	1	4	(0/0)	(0/0)
5	2	1	4	(0/0)	(0/0)
6	3	1	4	(0/0)	(0/0)
7	3	1	4	(0/0)	(0/0)
6	3	2	8	(0/0)	(0/0)
7	3	2	4	(0/0)	(0/0)

Table 6. Results for (no own / one opponent) memory, if strategies do not play against themselves. The numbers in parentheses in the "First It." column denote period length, if the results oscillate. If a rule is only among the winners of the tournament every other iteration, then it is displayed in italics. This setting is much less prone to lead to cooperation than if strategies also play against themselves.

Strategy Number	Latest Own	Latest Opponent
(?/1)	D	D
(?/2)	C	D
(?/4)	D	C
(?/8)	C	C

Table 7. A strategy cooperates if its number is composed of elements from this table. For example, the strategy TFT is (1/12) (cooperate, if line three or line four is remembered: (1/4+8)).

In Table 8 the set of 4 consists of the strategies (0/0), (0/2), (0/8), and (0/10). All winning strategies cooperate in the first iteration and at least continue to cooperate upon mutual cooperation (1/≥8). If $b > a + 1$ then, (1/12) (TFT) is not among the winners. Strategy (?/9) continues its behavior if the opponent has cooperated, or changes it, that is, it is Pavlovian. (1/8) can also be seen as a Pavlovian strategy, but somewhat more content than (1/9). It is happy with anything other than S and thus repeats its previous behavior unless it receives an S. If the opponent defects, no cooperating rule is among the winners. (Strategy (0/2) would do so, but never reaches a cooperative state.)

T	R	P	First It.	G_i^1	Tournament
3	2	1	8	set of 4	(1/8), (1/12)
4	3	1	66	(1/8)	(1/8), (1/9), (1/12), (1/13)
5	3	2	18	set of 4	(1/8), (1/12)
5	4	2	21	(1/8)	(1/8), (1/12)
6	5	2	18	(1/8)	(1/8), (1/9), (1/12), (1/13)
4	3	2	12	set of 4	(1/8), (1/12)
6	4	3	21	set of 4	(1/8), (1/12)
6	5	3	27	(1/8)	(1/8), (1/12)
5	3	1	8	set of 4	(1/8), (1/12)
7	4	2	15	set of 4	(1/8), (1/12)
9	5	3	18	set of 4	(1/8), (1/12)
4	2	1	1398	set of 4	(1/12)
5	2	1	10	set of 4	(1/8)
6	3	1	30	set of 4	(1/8), (1/12)
7	3	1	6	set of 4	(1/8)
6	3	2	645	set of 4	(1/12), *(1/8)*
7	3	2	15	set of 4	(1/8)

Table 8. Results for (one own / one opponent) memory, if strategies also play against themselves.

T	R	P	First It.	G_i^1	Tournament
3	2	1	34	set of 4	(1/8), (1/12)
4	3	1	29	(1/8)	(1/8), (1/12)
5	3	2	30	set of 4	(1/8), (1/12)
5	4	2	42	(1/8)	(1/8), (1/12)
6	5	2	18	(1/8)	(1/8), (1/12)
4	3	2	23	set of 4	(1/8), (1/12)
6	4	3	39	set of 4	(1/8), (1/12)
6	5	3	53	(1/8)	(1/8), (1/12)
5	3	1	363	set of 4	(1/8), (1/12)
7	4	2	57	set of 4	(1/8), (1/12)
9	5	3	163	set of 4	(1/12), *(1/8)*
4	2	1	49	set of 4	set of 4
5	2	1	69	set of 4	set of 4
6	3	1	9	set of 4	set of 4
7	3	1	7	set of 4	set of 4
6	3	2	66	set of 4	set of 4, *(0/4)*
7	3	2	141	set of 4	set of 4 altern. ((0/4), (1/4))

Table 9. Results for (one own / one opponent) memory, if strategies do not play against themselves. The set of 4 consists of the strategies (0/0), (0/2), (0/8), and (0/10).

6.3 Results for No-Own-Two-Opponent Memory

We used 10 000 iterations for this configuration. Again, this is far more than the largest number of iterations before the process settles down in some way. Now TFT is (1/2/12) and tit-for-two-tat (TF2T) is (1/3/14). Results are shown in Tables 10 and 11.

Table 10 shows that for 6-3-1 strategy (1/0/2) wins for two iterations and then (0/1/2) and (0/3/2) win. For 7-3-1 it is similar, but strategy (0/3/2) never wins. Compared to Table 5 TFT (1/2/12) (or even more cooperative strategies) mostly reappears, only disappears as winner of the tournament for 6-5-3, but newly wins 9-5-3. Thus, the general tendency that payoff matrices with $b \leq 1$ produce more cooperation is kept, but is less pronounced. The most cooperative strategy to co-win a tournament is (1/3/14), which only defects if it remembers two defections of the opponent. Overall—compared to the settings with smaller memory—the dominance of ALLD has vanished, especially in the first round of the tournament.

T	R	P	First It.	G_i^1	Tournament
3	2	1	383	(1/2/2)	(1/2/10), (1/3/10), (1/2/12), (1/3/12)
4	3	1	350	(1/2/2)	(1/3/10), (1/2/14)
5	3	2	179	(0/0/2)	(1/2/10), (1/3/10), (1/2/12), (1/3/12)
5	4	2	422	(1/2/2)	(1/2/10), (1/3/10), (1/2/12), (1/3/12), (1/2/14), (1/3/14)
6	5	2	397	(1/2/2)	(1/2/10), (1/3/10), (1/2/12), (1/3/12), (1/2/14), (1/3/14)
4	3	2	53	(0/0/0)	(1/2/10), (1/3/10), (1/2/12), (1/3/12), (1/2/14), (1/3/14)
6	4	3	35	(0/0/0)	(1/2/8), (1/3/8), (1/2/10), (1/3/10), (1/2/12), (1/3/12)
6	5	3	1076	(0/0/0)	(1/3/10)
5	3	1	215	(1/2/2)	(0/3/2)
7	4	2	527	(1/2/2)	(0/3/2)
9	5	3	2123	(1/2/2)	(1/3/10), (1/3/12)
4	2	1	719 (2)	(1/2/2)	(1/2/4) altern. (0/3/4)
5	2	1	1283 (2)	(0/0/2)	(0/2/4) altern. (0/3/4)
6	3	1	299 (4)	(1/2/2)	(1/0/2)
7	3	1	395 (4)	(1/2/2)	(1/0/2)
6	3	2	41 (2)	(0/0/2)	(1/2/4) altern. (0/3/4)
7	3	2	127 (2)	(0/0/2)	(1/2/4) altern. (0/3/4)

Table 10. Results for (no own / two opponent) memory, if strategies also play against themselves.

Table 11 shows that for payoff matrices 7-4-2 and 9-5-3 strategy (0/2/4) co-wins in two out of three rounds. The comparison to Table 6 reveals that increasing memory size makes cooperative strategies much more successful for almost all payoff matrices. None of the payoff matrices that produced oscillating results with size-one memory do so with size-two memory and vice versa.

T	R	P	First It.	G_i^1	Tournament
3	2	1	959	(1/2/2)	(1/3/10), (1/3/12)
4	3	1	219	(1/2/2)	(1/3/10), (1/3/12), (1/3/14)
5	3	2	179	(0/0/2)	(1/3/10), (1/3/12)
5	4	2	720	(1/2/2)	(1/3/10)
6	5	2	619	(1/2/2)	(0/3/14)
4	3	2	276	(0/0/0)	(1/2/10), (1/3/10), (1/2/12), (1/3/12), (1/2/14), (1/3/14)
6	4	3	38	(0/0/0)	(1/2/8), (1/3/8), (1/2/10), (1/3/10), (1/2/12), (1/3/12)
6	5	3	422	(0/0/0)	(1/3/10), (1/0/12), (0/3/14)
5	3	1	359	(1/2/2)	(0/3/2)
7	4	2	1224 (3)	(0/0/2)	(1/2/4), *(0/2/4)*
9	5	3	1644 (3)	(0/0/2)	(1/2/4), *(0/2/4)*
4	2	1	2891 (2)	(0/0/2)	(0/2/4), *((1/2/4) alt. (0/3/4))*
5	2	1	13 (2)	(0/0/2)	(0/2/4), *(0/3/4)*
6	3	1	515 (4)	(1/2/2)	(1/0/2)
7	3	1	731	(1/2/2)	(1/0/2)
6	3	2	85 (2)	(0/0/2)	(0/2/4), *((1/2/4) alt. (0/3/4))*
7	3	2	115 (2)	(0/0/2)	(0/2/4), *((1/2/4) alt. (0/3/4))*

Table 11. Results for (no own / two opponent) memory, if strategies do not play against themselves.

6.4 Results for One-Own-Two-Opponent Memory

In this case, the size of the strategy can be reduced because there is no need to distinguish between strategies that cooperate or defect in the second iteration, if hypothetically they cooperated in the first iteration, when in fact they defected. The number of strategies was not reduced to the subset of distinguishable ones for this simulation. Doing so would have introduced an error in the source code, and at this stage, the effect on required computational resources is negligible. Thus, for each strategy there are three more that yield exactly the same results against each of the strategies. Just the smallest of the four equivalent strategies is given in Table 12. This means that in the case of initial defection adding 2, 8, or 10 to the middle number gives the equivalent strategies and in the case of initial cooperation, it is 1, 4, or 5. Therefore, the TFT strategy is (1/8/240), (1/9/240), (1/12/240), and/or (1/13/240). Even when the results are reduced by naming only one of four strategies linked in this way, this is the first configuration that is too complicated to be understandable at a glance.

In Table 12 the "∨" is used as the common meaning of "or". (1/10/160) cooperates in the first and second iterations and then continues to cooperate, if both strategies have cooperated, otherwise it defects. This implies that it does not make use of the information of the second-to-last iteration and is therefore simpler than possible. Except for that it always cooperates in the second iteration, it is strategy (1/8) from the (one / one) setting. The set of 4 strategies consists of (0/0/1∨9∨129∨137), which all use information about the opponent's second-to-last action. The set of 22 is (1/8∨10/176∨180∨208∨212∨240∨244), (1/8/144∨146∨148∨150∨178∨182∨210∨214∨242∨246) and includes TFT. The set of 13 is (1/10/148), (1/8∨10/132∨140∨164∨196∨204∨228). The set of 17 includes the set of 13, (1/8/168∨172∨232), and (1/10/144). The set of 30 contains the set of 13, (1/8∨/10/128∨136∨160∨192∨200∨224), (1/8/130∨162∨194∨226), and (1/10/144). The set of 37 consists of the set of 30, (1/8∨10/168∨172∨232), and (1/10/236). The remaining four sets (20, 39, 25, and 29) share in common (1/10/168∨172∨184∨188∨204∨232∨236∨248∨252), including TF2T. A total of 41 further strategies appear as members of these sets, of which a majority (28) have been omitted from the table.

There are even more strategies that yield identical results when combined with any other player. For all strategies that continue to defect (cooperate) after an initial defection (cooperation), there are elements that determine what to do following a cooperation (defection). These elements are never applied and their values have no effect. This phenomenon leads to a large number of winning strategies. Interestingly, for some of the payoff matrices the number of winners is smaller—around 20 or 30 iterations—than at larger numbers of iterations.

For this memory configuration there is almost no difference in the results whether strategies play against themselves or not. The strategies with the most points in the first round of the tournament and the number of strategies winning the tournament are the same in both cases. If the number of winning strategies is large, a small number of strategies might be exchanged, causing differences in the iteration round when results become stable. In iteration rounds before stability, there can be larger differences, however. We refrain from giving a table of the results for the case when strategies do not play against themselves.

T	R	P	First It.	G_i^1	Tournament
3	2	1	1436	set of 4	set of 22, set of 17
4	3	1	998	set of 4	set of 22, set of 20
5	3	2	134	set of 4	set of 22, set of 13
5	4	2	234	set of 4	set of 22, set of 19
6	5	2	804	(1/10/160)	set of 22, set of 39
4	3	2	1838	set of 4	set of 22, set of 37
6	4	3	794	set of 4	set of 22, set of 30
6	5	3	929	(1/10/160)	set of 22, set of 25
5	3	1	2188	set of 4	set of 22, (1/10/148)
7	4	2	39	set of 4	set of 22, (1/10/148)
9	5	3	45	set of 4	set of 22, (1/10/148)
4	2	1	412	set of 4	(0/1∨5/180∨244), (0/5/176∨244)
5	2	1	278	set of 4	(0/1/180)
6	3	1	133 (2)	set of 4	(0/1∨5/180∨244), (0/5/176∨244), *(0/1/244)*
7	3	1	2174	set of 4	(0/1/180)
6	3	2	324	set of 4	(0/1∨5/180∨244), (0/5/176∨244)
7	3	2	422	set of 4	(0/1/180)

Table 12. Results for (one own / two opponent) memory, if strategies also play against themselves.

6.5 Results for Two-Own-One-Opponent Memory

This configuration is interesting because a strategy considers an opponent's action as a reaction to its own remembered action. While TFT is (1/8/240), a strategy that also cooperates in this case would be (1/8/244). As Table 13 shows, sometimes only TFT appears among the winners of the tournament, sometimes both of these strategies. Only with payoff matrix 6-5-2 does the more forgiving strategy win, but not TFT. It is the more tricky strategy (1/8/228) that applies this kind of forgiveness, which is more successful than TFT.

In this setting as well, if a strategy plays against itself or not has only minor effects. Therefore, the results for the case when they do not is omitted.

In Table 13 for the payoff matrices from the top down to 5-3-1 strategy (1/8/228) is always among the winners. This strategy almost always plays tit-for-tat, but does not cooperate if the opponent has cooperated and it has defected two times itself. However, it does cooperate if the opponent has defected after it has defected, even if it has cooperated in the most recent game. The history for the winning strategy (0/1/4) of the first round of the tournament shows that is the only case when it cooperated. 20 000 iterations were calculated for the payoff matrices 5-3-2 and 7-4-2 to verify the late stability, respectively period 4.

T	*R*	*P*	First It.	G_i^1	Tournament
3	2	1	539 (2)	(0/1/4)	(1/8∨10/164∨228),
					(1/10/set of 13 altern. set of 14)
4	3	1	338	(0/1/4)	(1/8/228∨229)
5	3	2	8367	(0/1/4)	(1/8/228)
5	4	2	107	(0/1/4)	(1/8∨10/224∨228∨240∨244),
					(1/10/set of 14)
6	5	2	111	(0/1/4)	(1/8/228∨229∨244)
4	3	2	3768	(0/1/4)	(1/8∨10/224∨228∨240),
					(1/10/set of 11)
6	4	3	242	(0/1/4)	(1/8∨10/164∨224∨228∨240),
					(1/10/set of 12)
6	5	3	483	(0/1/4)	(1/8/224∨228∨240∨244)
5	3	1	106	(0/1/4)	(1/8∨10/164∨228),
					(1/10/160∨161∨176∨177∨224∨225∨240∨241)
7	4	2	5989 (4)	(0/1/4)	(1/8∨10/160∨176∨224), (1/10/240)
9	5	3	350	(0/1/4)	(1/8∨10/160∨176∨224), (1/10/240)
4	2	1	32 (2)	(0/1/4)	(1/8/224∨*160∨176*)
5	2	1	32	(0/1/4)	(0/5/224)
6	3	1	407 (2)	(0/1/4)	(1/8/160∨161∨176∨177∨224∨225),
					altern. (0/5/224∨225)
7	3	1	29	(0/1/4)	(0/5/224∨225)
6	3	2	37 (2)	(0/1/4)	(1/8/160∨176∨224), altern. *(0/5/224)*
7	3	2	35	(0/1/4)	(0/5/224)

Table 13. Results for (two own / one opponent) memory, if strategies also play against themselves.

6.6 Results for No-Own-Three-Opponent Memory

This setting has the largest number of strategies investigated in this paper. The number of iterations until the results settle varies greatly among the various payoff matrices. In fact, for some payoff matrices they did not stabilize before iteration 30 000. At that point we refrained from further calculations and accepted the (non-)result as an open issue for future investigations. However, even for payoff matrices that have reached apparently stable results it cannot be excluded that after some 10 000 further iterations more different winners would result, as in the more volatile cases. Another surprising observation was that the results sometimes appeared to have reached a final state but then started changing again. After all, for remembering one opponent's action, stable results appeared after approximately 10 iterations, and for remembering two opponents' moves it was about 1000 iterations. So, it is not unrealistic to assume that remembering three opponents' actions may need 100 000 or even more iterations until the results do not change anymore.

Further difficulties may arise from precision issues in the calculation. During the tournament, which strategies may participate in the next round is decided by comparing the average of their points. The average is calculated by dividing one very large number by another very large number. As a consequence, the size comparison between average and individual results may be faulty. In fact, if a strategy has exactly achieved the average of points it is kicked out of the tournament. Another possible resource problem is that the sum of points may produce an overflow in the corresponding integer variable. Such considerations are generally known to be relevant when dealing with such large numbers during complex simulations. There was no explicit hint in our results that such issues really occurred, except the surprisingly long instability of results that could, in principle, be attributed to them. Ruling those considerations out would require a second computer system with a different architecture or a very thorough understanding of the CPU and the compiler being used. None of these were sufficiently available. Additionally, each simulation run currently takes days to arrive at the number of iterations where these issues could be relevant. When using up-to-date standard computer systems the no-own-three-opponent-memory case is at the edge of accessibility. Definitely ruling out negative effects that falsify the results with a maintainable effort remains to be done in the future.

Calculating the payoff and evaluating the tournament takes more computation time than calculating the results of the dilemma. Therefore, payoff calculation and tournament evaluation were only carried out for the last 100 iterations before each full 1000^{th} iteration if there were more than 10 000 iterations in total. This in turn implies that we can only approximate the iteration round after which the results are stable.

Having said all this, it becomes obvious that the results of this section need to be considered as preliminary—even more so the later the assumed stability was observed.

A different problem is that in some cases the number of tournament winners is too large to give all of them in this paper. However, the remaining cases should be sufficient to demonstrate the types and variants of strategies that win.

A majority of strategies that win the first round of a tournament cooperate when the earliest remembered opponent's action was cooperation, and any other defection. This trend was already present with the two-opponent-memory, but it was not as pronounced. This strategy is interesting because it uses the last chance to avoid breaking entirely with the opponent. To find a catchy name for this strategy, recall Mephisto's behavior toward God in the "Prologue in Heaven" of *Faust I*: "The ancient one I like sometimes to see, And not to break with him am always civil," where even considering all the competition between the two, Mephisto avoids entirely abandoning coopera-