Complete Classes of Strategies for the Classical Iterated Prisoner's Dilemma

Bruno BEAUFILS, Jean-Paul DELAHAYE, and Philippe MATHIEU

Laboratoire d'Informatique Fondamentale de Lille
U.R.A. 369 C.N.R.S. – Université des Sciences et Technologies de Lille
U.F.R. d'I.E.E.A. Bât. M3
59655 Villeneuve d'Ascq Cedex – FRANCE
{beaufils,delahaye,mathieu}@lifl.fr

Abstract. The Classical Iterated Prisoner's Dilemma (CIPD) is used to study the evolution of cooperation. We show, with a genetic approach, how basic ideas could be used in order to generate automatically a great numbers of strategies. Then we show some results of ecological evolution on those strategies, with the description of the experimentations we have made. Our main purpose is to find an objective method to evaluate strategies for the CIPD. Finally we use the former results to add a new argument confirming that there is, in order to be *good*, an infinite gradient in the level of complexity in structure of strategies.

1 The Classical Iterated Prisoner's Dilemma

Introduced by Merill M. FLOOD and Melvin DRESHER in the RAND Corporation in 1952, see [3], who tried to introduce some *irrationality* in the game theory of John VON NEUMANN and Oskar MORGENSTERN [8], the Classical Iterated Prisoner's Dilemma (CIPD) is based on this simple story quoted by Albert TUCKER for instance in [5, pages 117–118]:

> *Two men, charged with a joint violation of law, are held separately by the police. Each is told that*
> *(1) if one confesses and the other does not, the former will be given a reward... and the latter will be fined*
> *(2) if both confess, each will be fined...*
> *At the same time, each has a good reason to believe that*
> *(3) if neither confesses, both will go clear.*

It seems clearly obvious that the most reasonable choice is to betray its partner. More formally the CIPD is represented, using game theory, as a two-person non-zero-sum non-cooperative and simultaneous game where each player has to choose between two moves:

- COOPERATE, let us write `C`, and let us say to be *nice*
- DEFECT, let us write `D`, and let us say to be *naughty*

Table 1. CIPD payoff matrix. Row player score are given first.

	Cooperate	Defect
Cooperate	$R = 3, R = 3$ *Reward* for mutual cooperation	$S = 0, T = 5$ *Sucker*'s payoff *Temptation* to defect
Defect	$T = 5, S = 0$ *Temptation* to defect *Sucker*'s payoff	$P = 1, P = 1$ *Punishment* for mutual defection

The payoff of each player depends on the moves played by the two people. Table 1 provides the score in each case.

To have a dilemma, the following inequation has to be respected:

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

The dilemma stands on the fact that individual interests differ from collective ones.

As the one shot game is solved by the NASH equilibrium, which is to always betray its partner, the model is extended: in the iterated version players meet each other more than one time, without knowing if it is the last time or not. The payoff of a player is then simply the sum of each of its meeting's payoff. To favour the cooperation, and also to keep this difference between individual and collective interest the following inequation has to be respected:

$$S + T < 2R \tag{2}$$

The classical choice of values for the four parameters is given in Table 1.

Of course, with such an iterated game, what the opponent did on the past moves may influence the way a player will choose their next one. It is then possible to define more strategies than in the one shot version.

Let us define some simples ones:

all_c corresponds to the C strategy of the one shot game applied without modifications in the CIPD: it always plays C

all_d corresponds to the D strategy of the one shot game applied without modifications in the CIPD: it always plays D

tit_for_tat cooperates on the first move and then plays its opponent's previous move.

per_cd plays periodically C then D, let us note (CD)$*$

soft_majo cooperates, then plays opponent's most used move, if equal then cooperates

prober plays (DCC), then it defects in all other moves if opponent has cooperated in move 2 and 3, and plays as tit_for_tat in other cases

spiteful cooperates until first opponent's defection, then always defects.

2 Round Robin Tournament and Ecological Evolution

Now the main problem is to evaluate strategies for the CIPD, in order to compare them. Two kinds of experimentation could be used for this purpose. The basic one, is to make a pairwise round-robin tournament between some different strategies. The payoff to each one would be the total sum of each iterated game. A ranking could then be computed according to the score of each strategy. The higher a strategy is ranked, the better it is. Good strategies in round-robin tournament are well adapted to their environments, but often are not very robust to environment modifications.

The second kind of experimentation is a kind of imitation of the natural selection process, and is closely related to population dynamics. Let us consider a population of N players, each one adopting a particular strategy. At the beginning we consider that each strategy is equally represented in the population. Then a tournament is made, and good strategies are favoured, whereas bad ones are disadvantaged, by a proportional population redistribution. This redistribution process, also called a generation, is repeated until an eventual population stabilisation, *i.e.* no changes between two generations. A good strategy is then a strategy which stays alive in the population for the longest possible time, and in the biggest possible proportion.

An example of evolution between all strategies described in the previous section is shown in Figure 1. The x-axis represents the generation number, whereas the y-axis represents the size of the population for each strategy. For simplicity we make our computation with a fixed global population size.

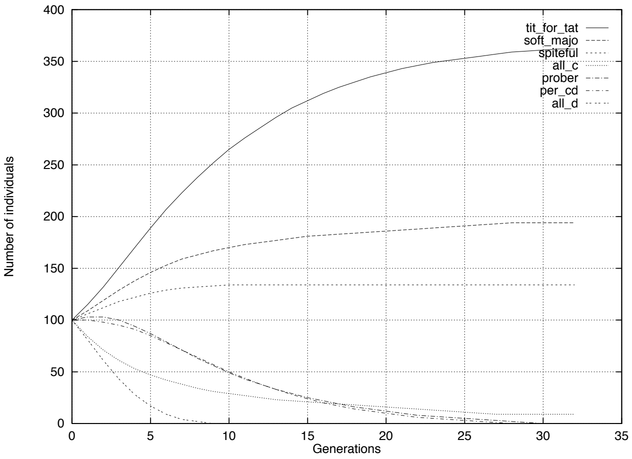

Fig. 1. Example of ecological evolution.

Classical results in this field, which were presented by AXELROD in [1], show that to be good a strategy must:

- be nice, *i.e.* not be the first to defect

– be reactive
– forgive
– not be too clever, *i.e.* to be simple in order to be understood by its opponent

The well-known `tit_for_tat` strategy, which satisfies all those criteria, has, since [1], been considered by a lot of people using the dilemma -but not by game theorist- to be one of the *best* strategies not only for cooperation but also for evolution of cooperation. We think that the *simplicity* criterium is not so good, and have thus introduced a strategy called `gradual` which illustrates our point of view.
`Gradual` cooperates on the first move, then after the first opponent's defection defects one time, and cooperates two times, after the second opponent's defection defects two times and cooperates two times, . . . , after the n^{th} opponent's defection defects n times, let us call it the punishment period, and cooperates two times, let us name it the lull time. `Gradual` had better results than `tit_for_tat` in almost all our experiments, see [2].
It is easy to imagine strategies derived from `gradual`, for instance in modifying the function of punishment, which is the identity in the `gradual` case (n defections $\Rightarrow$ punishment of length n).
Our main research direction to establish our ideas about a strategy's complexity is to

– try a lot of different strategies, in an automatic and objective manner
– have a general and, as far as possible, objective method to evaluate strategies, for instance to compare them.

3 Complete Classes of Strategies

To make our research process easier, we have to find a *descriptive* method to define strategies, which is less risky than an exhaustive method, which are never objective, nor complete. One of the ways we have chosen is to use a genetic approach. We describe a structure (a *genotype*), which can be decoded in a particular behaviour (a *phenotype*). Then a way to have a strategy, is simply to fill this structure. A manner to have a lot of strategies is to consider all the ways of filling this genotype, *i.e.* to consider all possible individuals based on a given genotype. Let us call the set of all strategies described by a particular genotype, the *complete class* of strategies issued by this genotype.
We have described three genotypes based on the same simple idea, in order to remain objective. This idea is to consider the observable length of the game's history. Such idea of strategies has already been studied in [4, 7]. Those three genotypes are:

`memory`: Each strategy can only see M_{ml} moves of its past, and O_{ml} moves of its opponent's past. The strategy is then started by $\max(M_{ml}, O_{ml})$ moves predefined in the genotype. All other moves are coded in the genotype, according to the visible past configuration. The genotype length is then $\max(M_{ml}, O_{ml}) + 2^{(M_{ml}+O_{ml})}$.

`binary_memory`: The same as the previous one, but the reply to an opponent's move depends not only on the visible past but also on the fact that past opponent's defection are more numerous, or not, than past cooperation. The genotype length is then $\max(M_{ml}, O_{ml}) + 2^{(M_{ml}+O_{ml}+1)}$.

memory_automata: Represents classical two-state automata, which starts in state 0. Each strategy can only see M_{ml} moves of its past, and O_{ml} moves of its opponent past. The strategy is then started by $\max(M_{ml}, O_{ml})$ moves predefined in the genotype. All other moves are coded in the genotype, according to the visible past configuration and the automata's current state. State transitions are also coded in the genotype. The genotype length is then $\max(M_{ml}, O_{ml}) + 2^{(M_{ml}+O_{ml}+2)}$. Strategies of such a kind, with $M_{ml} = O_{ml} = 3$, have already been studied in [6].

Despite the apparently simplicity of the strategies described by those genotypes, many classical ones are included in those classes, tit_for_tat for instance.
Let us describe the behavior of one of those strategies, in order to understand how the genotype works. We consider a strategy of the complete class of memory with $M_{ml} = 1$ and $O_{ml} = 1$.
One of the individuals of this class plays C on the first move then:
- if on the previous move I played C and the opponent played C then I play C
- if on the previous move I played C and the opponent played D then I play D
- if on the previous move I played D and the opponent played C then I play D
- if on the previous move I played D and the opponent played D then I play D

The genotype of the strategy is then: $\boxed{\texttt{C}}\boxed{\texttt{C}}\boxed{\texttt{D}}\boxed{\texttt{D}}\boxed{\texttt{D}}$
This strategy is one way of coding spiteful.

4 Some Experiments

We have conducted some experiments using those complete classes. The main purpose was to evaluate other strategies in big ecological evolution, but also to try to find some *new good* strategies. In all our experiments we have computed one ecological evolution between all strategies of a class, and then another with gradual added to the population. Thus we have been able to partially confirm our ideas on the strength of this strategy, as shown in Table 2. In all the results of Table 2, gradual is better evaluated than tit_for_tat. We, however, do not show results of the evaluation of tit_for_tat here since it is included in some of the classes explored, thus its evaluation is only partial.

4.1 Some Memory and Binary_memory Classes

We performed many experiments on those classes, since they are the most simple and objective.
Evolution of two classes are presented on Figures 2 and 3.
Noticed that when $M_{ml} \geq 1$ or $O_{ml} \geq 1$, the population is no longer uniform since, there are more *naughty* strategies than *nice* ones, due to the starting moves.
We then compute some subclasses, including the same number of *naughty* than *nice* strategies. In order to create such classes we limit the starting moves to those containing only C moves or only D moves.
In almost all cases when we add gradual in those kinds of population it is well evaluated too.

Table 2. Some results of the evaluation of `gradual` in complete classes. *Class size* is the number of described strategies, whereas *evaluation* is the rank of the strategy at the end of an evolution of the complete class.

	M_{ml}	O_{ml}	class size	evaluation		
				`gradual`	`tit_for_tat`	`spiteful`
`memory`	0	1	8	1	1	1
	0	2	64	5	2	21
	1	1	32	2	3	1
	1	2	1024	6	13	37
`binary_memory`	0	1	32	1	2	1
	1	1	512	1	7	13
`memory_automata`	0	1	512	1	31	32

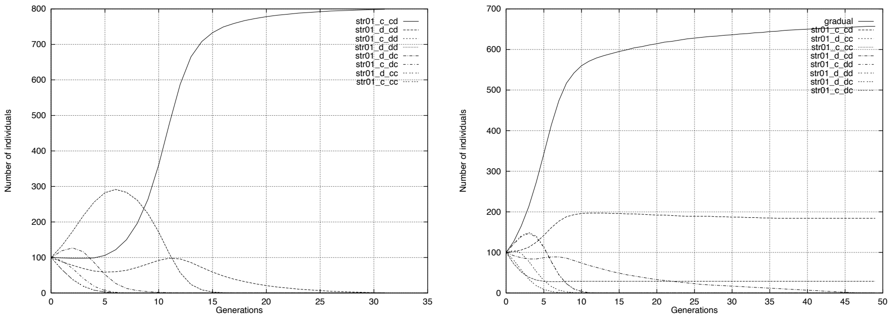

Fig. 2. Evolution of class `memory` ($M_{ml} = 0$ and $O_{ml} = 1$). Strategies issued from such complete classes are named `str`$M_{ml}O_{ml}$`_`<*genotype*>.

When `gradual` is not the winner of the evolution, we almost always find a *variation* of it, like `gradual_n4` (n defections ⇒ punishment of length n^4)) which does win. It is not exactly the case when those `gradual`'s variation are added in complete classes as shown in Figure 3.

One example of evolution of a `binary_memory` class is given in Figure 4.

4.2 A `Memory_automata` class

Due to the explosion of `memory_automata` class size relatively to parameters M_{ml} and O_{ml}, we just present the result of one of those classes, the case with $M_{ml} = 0$ and $O_{ml} = 1$ which contains 512 strategies.

As we can see on Figure 5 the best strategy of the `memory_automata` class with $M_{ml} = 0$ and $M_{ml} = 1$, is a strategy we called `str01e_c_0111_cccd`, which has the following genotype $\boxed{\mathtt{C}}\boxed{\mathtt{0}}\boxed{\mathtt{1}}\boxed{\mathtt{1}}\boxed{\mathtt{1}}\boxed{\mathtt{C}}\boxed{\mathtt{C}}\boxed{\mathtt{C}}\boxed{\mathtt{D}}$

The behavior of the strategy is then:

It begins (in state 0) to cooperate (playing `C`) then:

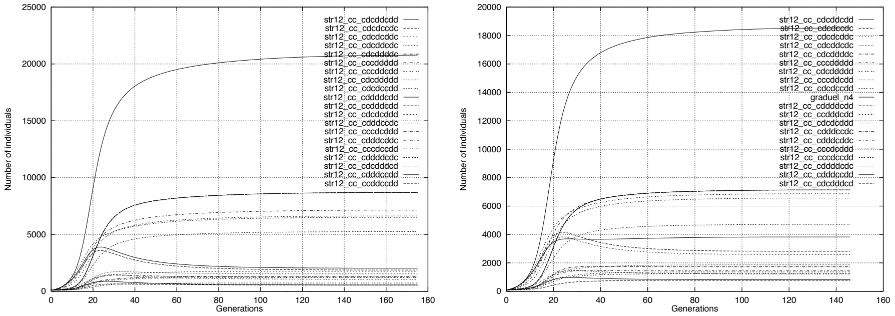

Fig. 3. Evolution of class `memory` ($M_{ml} = 1$ and $O_{ml} = 2$). Strategies issued from such complete classes are named `str`$M_{ml}O_{ml}$_<*genotype*>.

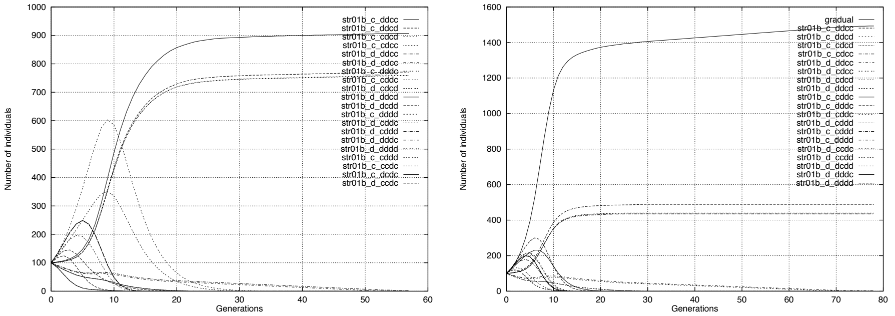

Fig. 4. Evolution of class `binary_memory` ($M_{ml} = 0$ and $O_{ml} = 1$). Strategies issued from such complete classes are named `str`$M_{ml}O_{ml}$`b`_<*genotype*>

- if I am in state 0 and opponent played `C` then I go to state 0 and play `C`;
- if I am in state 0 and opponent played `D` then I go to state 1 and play `C`;
- if I am in state 1 and opponent played `C` then I go to state 1 and play `C`;
- if I am in state 1 and opponent played `D` then I go to state 1 and play `D`;

This automata is represented by the scheme of Figure 6.
This strategy, which acts in a manner very close to the one of `tit_for_tat` is beaten by `gradual`, when the latter is added in the population.

5 Conclusions

The genetic method we use to define strategies for the CIPD offers two big advantages:

- It is easy to define a large number of strategies ;
- The way the strategies are defined is objective, *i.e.* without biased choice, so that it could be used to be part of the evaluation of a strategy.

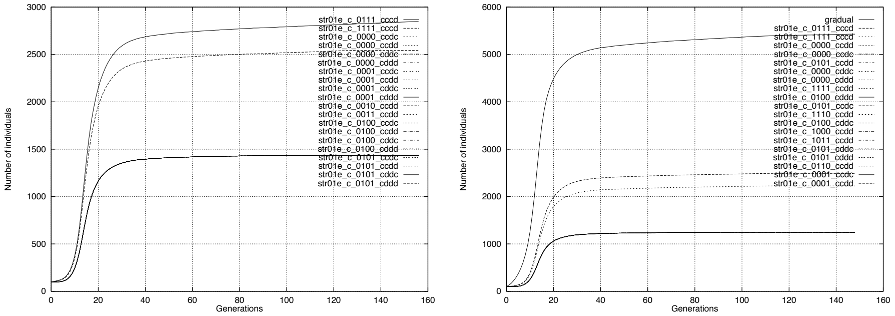

Fig. 5. Evolution of class `memory_automata` ($M_{ml} = 0$ and $O_{ml} = 1$). Strategies issued from such complete classes are named `str`$M_{ml}O_{ml}$`e_`<*genotype*>.

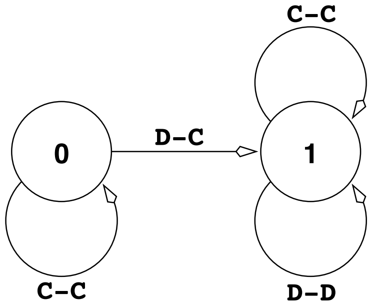

Fig. 6. Best strategy of class `memory_automata` ($M_{ml} = 0$ and $O_{ml} = 1$). Transitions on the arrows specify the opponent's last move and the reply of the strategy.

Evaluation of strategies based on complete classes cannot be considered as subjective as long as the genotype used is sufficently general, and based on basic ideas. Evaluation can, however, not be based only on the results of complete classes evolution, since a strategy could have a behavior well adapted to this kind of environment, and not well adapted to a completely different environment, with mixed strategy, in the game theory sense, for instance.

The results we obtained mainly confirm our ideas about a strategy's complexity, which is, there may exist an infinite gradient of complexity in the definition of strategy, each level defining a new criterium of quality.

We plan to include this kind of evaluation, in a more general, objective and complete evaluation method. We also plan to explore larger classes of strategy, which implies to find optimization method of computation, enabling us to describe and to use very large sets of strategies.

A simulation software with many strategies is already available for Unix, DOS or Windows on the World Wide Web at `http://www.lifl.fr/˜mathieu/ipd` or by anonymous ftp on the site `ftp.lifl.fr` in `pub/users/mathieu/soft`.

References

1. R. Axelrod. *The Evolution of Cooperation*. Basic Books, New York, USA, 1984.
2. B. Beaufils, J. Delahaye, and P. Mathieu. Our meeting with gradual, a good strategy for the iterated prisoner's dilemma. In C. G. Langton and K. Shimohara, editors, *Artificial Life V: Proceedings of the Fifth International Workshop on the Synthesis and Simulation of Living Systems*, pages 202–209, Cambridge, MA, USA, 1996. The MIT Press/Bradford Books.
3. M. M. Flood. Some experimental games. Research memorandum RM-789-1-PR, RAND Corporation, Santa-Monica, CA, USA, June 1952.
4. K. Lindgren. Evolutionary phenomena in simple dynamics. In C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, editors, *Artificial Life II: Proceedings of the Second Interdisciplinary Workshop on the Synthesis and Simulation of Living Systems*, volume 10 of *Santa Fe Institute Studies in the Sciences of Complexity*, pages 295–312, Reading, MA, USA, 1992. Addisson–Wesley.
5. W. Poundstone. *Prisoner's Dilemma : John von Neumann, Game Theory, and the Puzzle of the Bomb*. Oxford University Press, Oxford, UK, 1993.
6. A. Salhi, H. Glaser, D. D. Roure, and J. Putney. The prisoner's dilemma revisited. Technical Report DSSE-TR-96-2, University of Southhampton, Department of Electronics and Computer Science, Declarative Systems and Software Engineering Group, Southampton, UK, March 1996.
7. T. W. Sandholm and R. H. Crites. Multiagent reinforcement learning in the iterated prisoner's dilemma. *BioSystems*, 37(1,2):147–166, 1996.
8. J. von Neumann and O. Morgenstern. *Theory of Games and Economics Behavior*. Princeton University Press, Princeton, NJ, USA, 1944.