An Experimental Study of N-Person Iterated Prisoner's Dilemma Games

Xin Yao and Paul J. Darwen

Department of Computer Science
University College, The University of New South Wales
Australian Defence Force Academy
Canberra, ACT, Australia 2600

Abstract. The Iterated Prisoner's Dilemma game has been used extensively in the study of the evolution of cooperative behaviours in social and biological systems. There have been a lot of experimental studies on evolving strategies for 2-player Iterated Prisoner's Dilemma games (2IPD). However, there are many real world problems, especially many social and economic ones, which cannot be modelled by the 2IPD. The n-player Iterated Prisoner's Dilemma (NIPD) is a more realistic and general game which can model those problems. This paper presents two sets of experiments on evolving strategies for the NIPD. The first set of experiments examine the impact of the number of players in the NIPD on the evolution of cooperation in the group. Our experiments show that cooperation is less likely to emerge in a large group than in a small group. The second set of experiments study the generalisation ability of evolved strategies from the point of view of machine learning. Our experiments reveal the effect of changing the evolutionary environment of evolution on the generalisation ability of evolved strategies.

1 Introduction

The 2-player Iterated Prisoner's Dilemma game (2IPD) is a 2×2 non-zerosum noncooperative game, where "non-zerosum" indicates that the benefits obtained by a player are not necessarily the same as the penalties received by another player and "noncooperative" indicates that no preplay communication is permitted between the players [1, 2]. It has been widely studied in such diverse fields as economics, mathematical game theory, political science, and artificial intelligence.

In the Prisoner's Dilemma, each player has a choice of two operations: either cooperate with the other player, or defect. Payoff to both players is calculated according to Figure 1. In the Iterated Prisoner's Dilemma (IPD), this step is repeated many times, and each player can remember previous steps.

While the 2IPD has been studied extensively for more than three decades, there are many real world problems, especially many social and economic ones, which cannot be modelled by the 2IPD. Hardin [3] described some examples of such problems. More examples can be found in Colman's book [1](pp.156–159). The n-player Iterated Prisoner's Dilemma (NIPD) is a more realistic and general

	Cooperate	Defect
Cooperate	R R	T S
Defect	S T	P P

Fig. 1. The payoff matrix for the 2-player prisoner's dilemma game. The values S, P, R, T must satisfy $T > R > P > S$ and $R > (S + T)/2$. In 2-player Iterated Prisoner's Dilemma (2IPD), the above interaction is repeated many times, and both players can remember previous outcomes.

game which can model those problems. In comparing the NIPD with the 2IPD, Davis *et al.* [4](pp.520) commented that

> The N-player case (NPD) has greater generality and applicability to real-life situations. In addition to the problems of energy conservation, ecology, and overpopulation, many other real-life problems can be represented by the NPD paradigm.

Colman [1](pp.142) and Glance and Huberman [5, 6] have also indicated that the NIPD is "qualitatively different" from the 2IPD and that "... certain strategies that work well for individuals in the Prisoner's Dilemma fail in large groups."

The n-player Prisoner's Dilemma game can be defined by the following three properties [1](pp.159):

1. each player faces two choices between cooperation (C) and defection (D);
2. the D option is dominant for each player, i.e., each is better off choosing D than C no matter how many of the other players choose C;
3. the dominant D strategies intersect in a deficient equilibrium. In particular, the outcome if all players choose their non-dominant C strategies is preferable from every player's point of view to the one in which everyone chooses D, but no one is motivated to deviate unilaterally from D.

Figure 2 shows the payoff matrix of the n-player game.

A large number of values satisfy the requirements of Figure 2. We choose values so that, if n_c is the number of cooperators in the n-player game, then the payoff for cooperation is $2n_c - 2$ and the payoff for defection is $2n_c + 1$. Figure 3 shows an example of the n-player game.

With this choice, simple algebra reveals that if N_c cooperative moves are made out of N moves of an n-player game, then the average per-round payoff a is given by:

Number of cooperators among the remaining $n-1$ players

		0	1	2		$n-1$
player A	C	C_0	C_1	C_2	$\cdots$	C_{n-1}
	D	D_0	D_1	D_2	$\cdots$	D_{n-1}

Fig. 2. The payoff matrix of the n-player Prisoner's Dilemma game, where the following conditions must be satisfied: (1) $D_i > C_i$ for $0 \leq i \leq n-1$; (2) $D_{i+1} > D_i$ and $C_{i+1} > C_i$ for $0 \leq i < n-1$; (3) $C_i > (D_i + C_{i-1})/2$ for $0 < i \leq n-1$. The payoff matrix is symmetric for each player.

Number of cooperators among the remaining $n-1$ players

		0	1	2		$n-1$
player A	C	0	2	4	$\cdots$	$2(n-1)$
	D	1	3	5	$\cdots$	$2(n-1)+1$

Fig. 3. An example of the N-player game.

$$a = 1 + \frac{N_c}{N}(2n-3) \tag{1}$$

This lets us measure how common cooperation was just by looking at the average per-round payoff.

There has been a lot of research on the evolution of cooperation in the 2IPD using genetic algorithms and evolutionary programming in recent years [7, 8, 9, 10, 11, 12]. Axelrod [7] used genetic algorithms to evolve a population of strategies where each strategy plays the 2IPD with every other strategy in the population. In other words, the performance or fitness of a strategy is evaluated by playing the 2IPD with every other strategy in the population. The environment in which a strategy evolves consists of all the remaining strategies in the population. Since strategies in the population are constantly changing as a result of evolution, a strategy will be evaluated by a different environment in every generation. All the strategies in the population are co-evolving in their dynamic environments. Axelrod found that such dynamic environments produced strategies that performed very well against their population. Fogel [11] described similar experiments, but used finite state machines to represent strategies and evolutionary programming to evolve them.

However, very few experimental studies have been carried out on the NIPD in spite of its importance and its qualitative difference from the 2IPD. This paper presents two sets of experiments carried out on the NIPD. We first describe our experiment setup in Section 2. Then we investigate the impact of the number of players in the Prisoner's Dilemma game on the evolution of cooperation in Section 3. We are mainly interested in two questions here: (1) whether cooperation can still emerge from a larger group, and (2) whether it is more difficult to evolve cooperation in a larger group. The evolution of strategies for the NIPD can be regarded as a form of machine learning using the evolutionary approach. An important issue in machine learning is generalisation. Section 4 of this paper discusses the generalisation issue associated with co-evolutionary learning and presents some experiments with different evolutionary environments. Finally, Section 5 concludes with some remarks and future research directions.

2 Experiment Setup

2.1 Genotypical Representation of Strategies

We use genetic algorithms to evolve strategies for the NIPD. The most important issue here is the representation of strategies. We will use two different representations, both of which are look-up tables that give an action for every possible contingency.

One way of representing strategies for the NIPD is to generalise the representation scheme used by Axelrod [7]. In this scheme, each genotype is a lookup table that covers every possible history of the last few steps. A history in such a game is represented as a binary string of ln bits, where the first l bits represent the player's own previous l actions (most recent to the left, oldest to the right), and the other $n-1$ groups of l bits represent the previous actions of the other players. For example, during a game of 3IPD with a remembered history of 2 steps, $n=3, l=2$, one player might see this history:

$n=3, l=2$: Example history `11 00 01`

The first l bits, `11`, means this player has defected (a "1") for both of the previous $l=2$ steps. The previous steps of the other players are then listed in order: the `00` means the first of the other players cooperated (a "0") on the previous l steps, and the last of the other players cooperated (0) on the most recent step, and defected (1) on the step before, as represented by `01`.

For the NIPD remembering l previous steps, there are 2^{ln} possible histories. The lookup table genotype therefore contains an action (cooperate "0" or defect "1") for each of these possible histories. So we need at least 2^{ln} bits to represent a strategy. At the beginning of each game, there are no previous l steps of play from which to look up the next action, so each genotype should also contain its own extra bits that define the presumed pre-game moves. The total genotype length is therefore $2^{ln} + ln$ bits. We will use this genotype for the first set of results below, Figure 5 through to Figure 8.

This Axelrod-style representation scheme, however, suffers from two disadvantages. First, it does not scale well as the number of players increases. Second, it provides more information than is necessary by telling which of the other players cooperated or defected, when the only information needed is how many of the other players cooperated or defected. Such redundant information had reduced the efficiency of the evolution greatly in our experiments with this representation scheme. To improve on this, we use a new representation scheme which is more compact and efficient.

In our new representation scheme, each individual is regarded as a set of rules stored in a look-up table that covers every possible history. As a game that runs for, say, 500 rounds would have an enormous number of possible histories, and as only the most recent steps will have significance for the next move, we only consider every possible history over the most recent l steps, where l is less than 4 steps. This means an individual can only remember the l most recent rounds. Such a history of l rounds is represented by:

1. l bits for the player's own previous l moves, where a "1" indicates defection, a "0" cooperation; and
2. another $l \log_2 n$ bits for the number of cooperators among the other $n - 1$ players, where n is the number of the players in the game. This requires that n is a power of 2.

For example, if we are looking at 8 players who can remember the 3 most recent rounds, then one of the players would see the history as:

History for 8 players, 3 steps: `001 111 110 101` (12 bits)

Here, the `001` indicates the player's own actions: the most recent action (on the left) was a "0", indicating cooperation, and the action 3 steps ago (on the right), was a "1", i.e., defection. The `111` gives the number of cooperators among the other 7 players in the most recent round, i.e., there were $111_2 = 7$ cooperators. The `101` gives the number of cooperators among the other 7 players 3 steps ago, i.e., there were $101_2 = 5$ cooperators. The most recent events are always on the left, previous events on the right.

In the above example, there are $2^{12} = 2048$ possible histories. So 2048 bits are needed to represent all possible strategies. In the general case of an n-player game with history length l, each history needs $l + l \log_2 n$ bits to represent and there are $2^{l+l\log_2 n}$ such histories. A strategy is represented by a binary string that gives an action for each of those possible histories. In the above example, the history `001 111 110 101` would cause the strategy to do whatever is listed in bit 1013, the decimal number for the binary 001111110101.

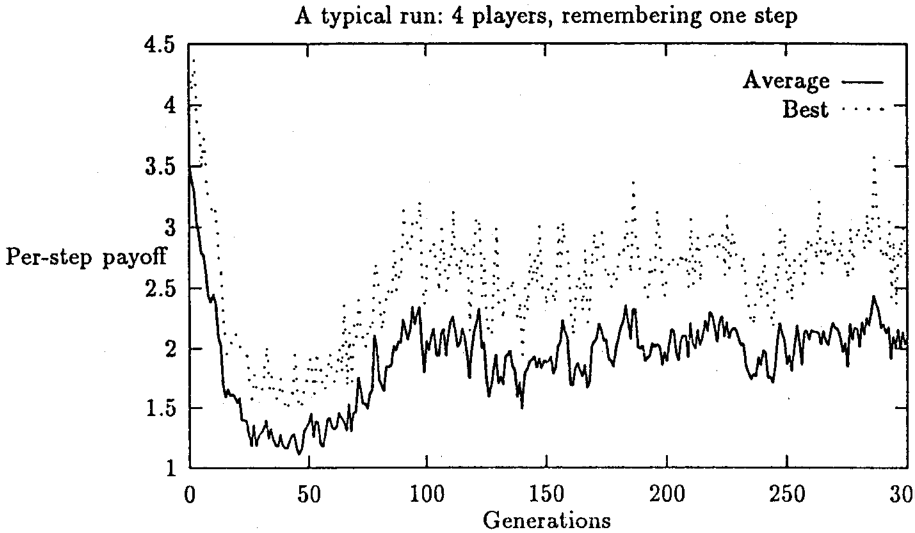

Fig. 4. This shows the average and best payoff at each generation for a population of 100 individuals. Each individual is a strategy.

Since there are no previous l rounds at the beginning of a game, we have to specify them with another $l(1 + \log_2 n)$ bits. Hence each strategy is finally represented by a binary string of length $2^{l+l \log_2 n} + l(1 + \log_2 n)$.

2.2 Genetic Algorithm Parameters

For all the experiments presented in this paper, the population size is 100, the mutation rate is 0.001, and the crossover rate is 0.6. Rank-based selection was used, with the worst performer assigned an average of 0.75 offspring, the best 1.25 offspring.

2.3 A Typical Run

A tyical run with four players with a history 1 ($n = 4, l = 1$) is shown in Figure 4. At each generation, 1000 games of the 4-player Iterated Prisoner's Dilemma are played, with each group of 4 players selected randomly with replacement. Each of these 1000 games lasts for 100 rounds. Starting from a random population, defection is usually the better strategy, and the average payoff plummets initially. As time passes, some cooperation becomes more profitable. We will examine more results in detail later.

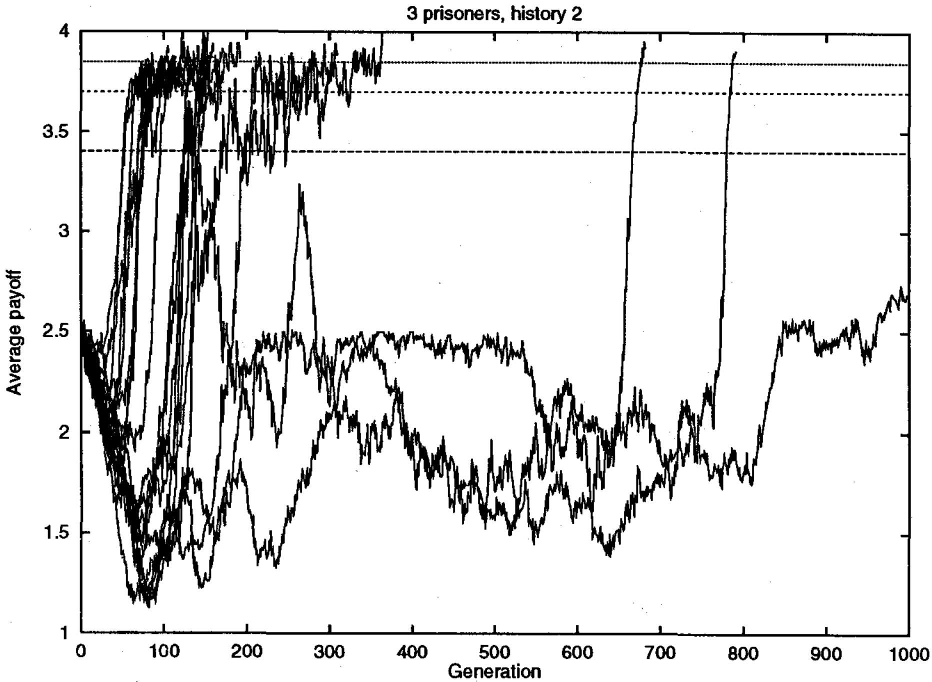

Fig. 5. For the 3-player prisoner's dilemma with a history of 2, cooperation almost always emerges. Only 1 out of 20 runs fail to reach 95% cooperation using Axelrod's representation shceme.

3 Group Size of the NIPD

This section discusses the impact of group size, i.e., the number of players in the NIPD, on the evolution of cooperation and presents some experimental results. It is well-known that cooperation can be evolved from a population of random strategies for the 2IPD. Can cooperation still be evolved from a population of strategies for the NIPD where the number of players is greater than 2? If the answer is yes, does the group size affect the evolution of cooperation in the NIPD?

Using the Axelrod-style genotype described above, we carried out a series of experiments with the 3IPD, 4IPD, 5IPD, and 6IPD games. In each of the following runs, the program stopped when more than 5 generations passed with the average payoff above the 95% cooperation level. Figure 5 shows the results of 20 runs of the 3IPD game with history length 2: out of 20 runs, there is only 1 which fails to reach 95% cooperation. Figure 6 shows the results of 20 runs of the 4IPD game with history length 2: 4 out of 20 runs fail to reach the 95% cooperation level, but only 1 of those fails to reach 80% cooperation. Figure 7 shows the results of 20 runs of the 5IPD game with history length 2: 6 out of 20 runs do not reach the 80% cooperation level. Figure 8 shows the results of 20

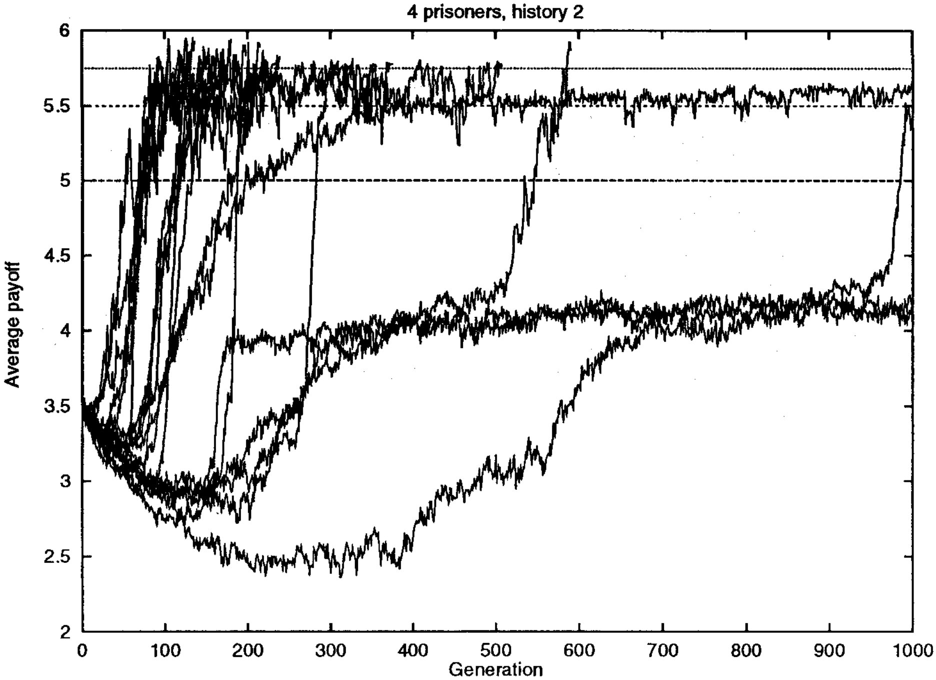

Fig. 6. For the 4-player prisoner's dilemma with a history of 2, cooperation almost always emerges. Only 4 out of 20 runs fail to reach 95% cooperation using Axelrod's representation shceme.

runs of the 6IPD game with history length 2: 9 out of 20 runs stay below the 80% cooperation level.

Figures 5 through 8 demonstrate that the evolution of cooperation becomes less likely as group size increases. Nonetheless, cooperation still emerges most of the time. As Axelrod's representation scheme used in those figures does not scale well with the group size, we use the second representation scheme described in Section 2 to carry out experiments with larger groups.

We have carried out a series of experiments with the 2IPD, 4IPD, 8IPD, and 16IPD games. Figure 9 shows the results of 10 runs of the 2IPD game with history length 3. Out of 10 runs, there are only 3 which fail to reach 90% cooperation and only 1 which goes to almost all defection. Figure 10 shows the results of 10 runs of the 4IPD game with history length 3, where some of the runs reach cooperation but more than half of the 10 runs fail to evolve cooperation. Figure 11 shows the results of 10 runs of the 8IPD game with history length 2, where none of the runs reach cooperation. Figure 12 shows the population bias in the runs in Figure 11, to demonstrate that those populations have pretty much converged. Figure 13 shows 10 runs of the 16IPD game.

These results confirm that cooperation can still be evolved in larger groups,

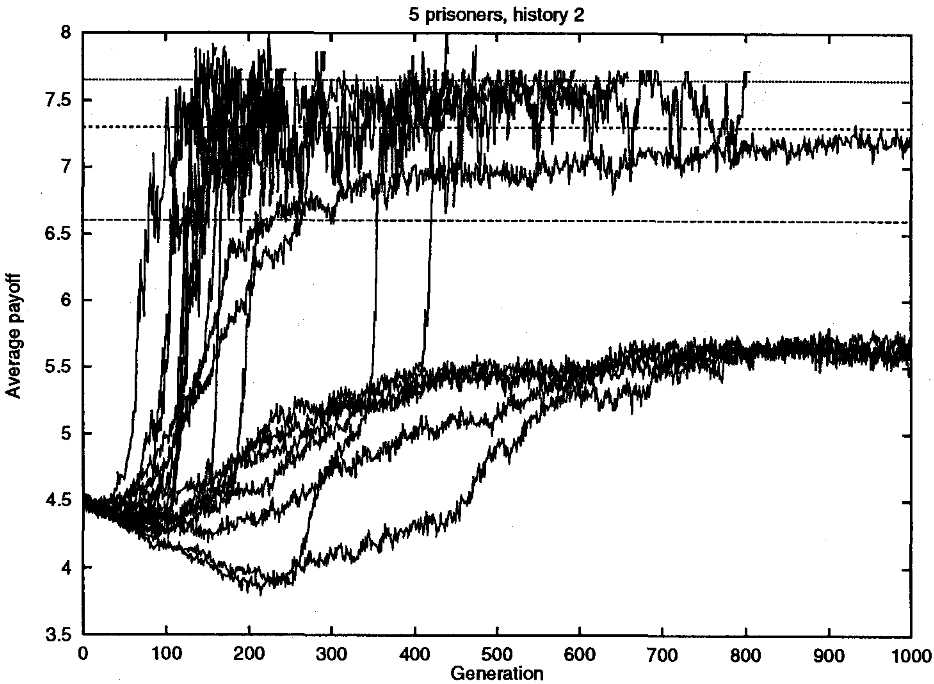

Fig. 7. For the 5-player prisoner's dilemma with a history of 2, cooperation almost always emerges. 6 out of 20 runs fail to reach 80% cooperation using Axelrod's representation shceme.

but it is more difficult to evolve cooperation as the group size increases. Glance and Huberman [5, 6] have arrived at a similar conclusion using a model based on many particle systems. We first suspected that the failure to evolve cooperation in larger groups was caused by larger search spaces and insufficient running time since more players were involved in 8IPD and 16IPD games. This is, however, not the case. The search space of the 8IPD game with history length 2 is actually smaller than that of the 4IPD game with history length 3. To confirm that the failure to evolve cooperation is not caused by insufficient running time, we examined the convergence of the 8IPD game. Figure 12 shows that at generation 200 the population has mostly converged for all the 10 runs.

It is worth mentioning that the evolution of cooperation using simulations does depend on some implementation details, such as the genotypical representation of strategies and the values used in the payoff matrix. So cooperation may be evolved in the 8IPD game if a different representation scheme and different payoff values are used. Although we cannot prove it vigorously, we think for any representation scheme and payoff values there would always be an upper limit on the group size over which cooperation cannot be evolved. Our experimental finding is rather similar to some phenomena in our human society, e.g., cooperation is usually easier to emerge in a small group of people than in a larger one.

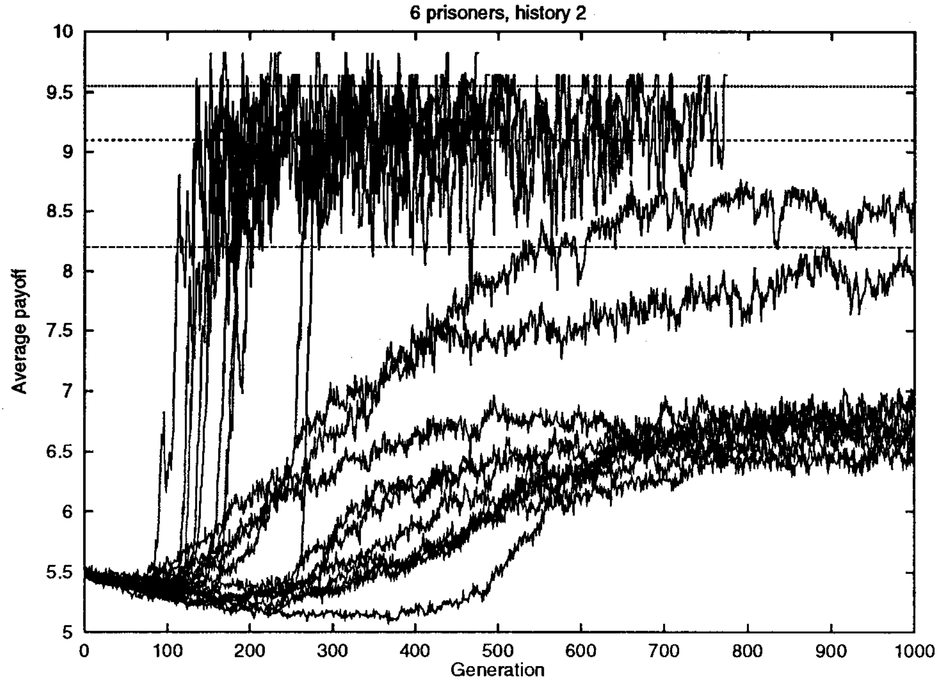

Fig. 8. For the 6-player prisoner's dilemma with a history of 2, cooperation almost always emerges. 9 1 out of 20 runs fail to reach 80% cooperation using Axelrod's representation shceme.

4 Co-Evolutionary Learning and Generalisation

The idea of having a computer algorithm learn from its own experience and thus create expertise without being exposed to a human teacher has been around for a long time. For genetic algorithms, both Hillis [13] and Axelrod [7] have attempted co-evolution, where a GA population is evaluated by how well it performs against itself or another GA population, starting from a random population. Expertise is thus bootstrapped from nothing, without an expert teacher. This is certainly an promising idea, but does it work? So far, no-one has investigated if the results of co-evolutionary learning are robust, that is, whether they generalise well? If a strategy is produced by a co-evolving population, will that strategy perform well against opponents never seen by that population? In order to investigate this issue, we need to pick the best strategies produced by the co-evolutionary learning system and let them play against a set of test strategies which had not been seen by the co-evolutionary system. This section describes some experiments which test the generalisation ability of co-evolved strategies for the 8IPD game with history length 1.

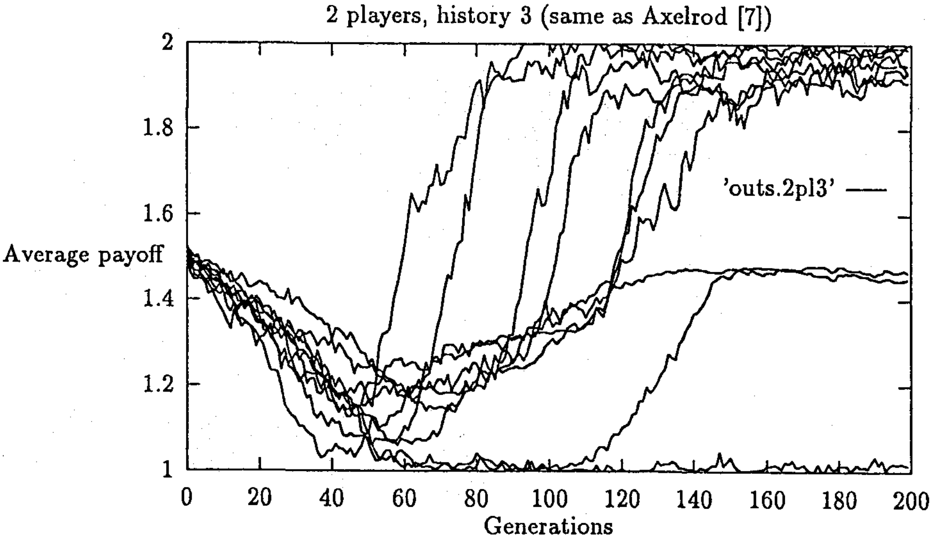

Fig. 9. For 2-player prisoner's dilemma with a history of 3, cooperation emerges most of the time. Only 3 out of 10 runs fail to reach 90% cooperation, and only 1 run goes to almost all defection.

4.1 Test Strategies

The unseen test strategies used in our study should be of reasonable standard and representative, that is, they are neither very poor (or else they will be exploited by their evolved opponents) nor very good (or else the will exploit their evolved opponent). We need unseen strategies that are adequate against a large range of opponents, but not *the* best.

To obtain such strategies, we did a limited enumerative search to find the strategies that performed best against a large number of random opponents. As most random opponents are very stupid, beating many random opponents provides a mediocre standard of play against a wide range of opponents. We limited this search to manageable proportions by fixing certain bits in a strategy's genotype that seemed to be sensible, such as always defecting after every other strategy defects. The top few strategies found from such a limited enumerative search are listed in Table 1.

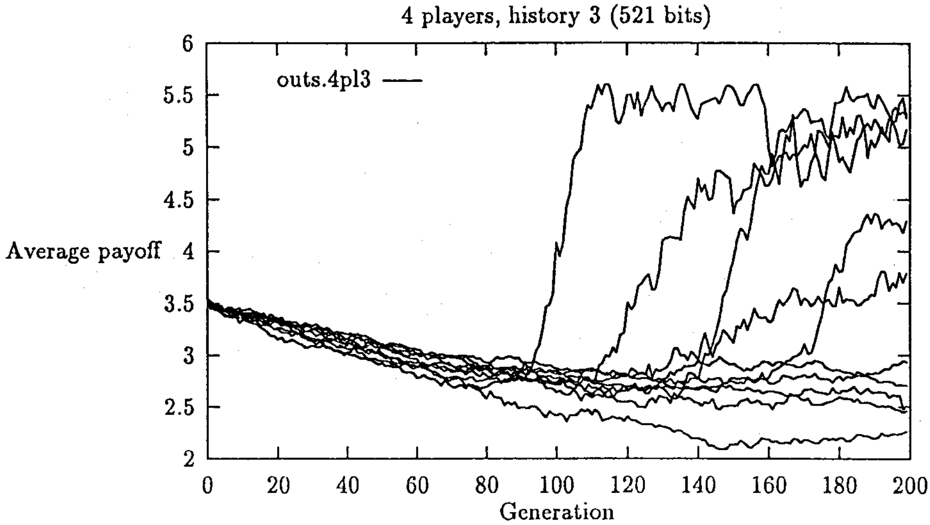

Fig. 10. For 10 runs of 4-player prisoner's dilemma with a history of 3, cooperation breaks out some of the time.

4.2 Learning and Testing

We have compared three different methods for implementing the co-evolutionary learning system. The three methods differ in the way each individual is evaluated, i.e., which opponents are chosen to evaluate an individual's fitness. The three methods are

1. Choosing from among the individuals in the GA population, i.e., normal co-evolution of a single population like Axelrod's implementation [7];
2. Choosing from a pool made of the evolving GA population and the best 25 strategies from the enumerative search, which remain fixed;
3. Choosing from a pool made of the evolving GA population and the best 25 strategies from the enumerative search, but the probability of choosing one of the 25 is four times higher.

For each of these, we obtained the best 25 strategies from the last generation of the GA, and tested it against a pool made up of both the seen and unseen enumerative search strategies, 50 in all.

4.3 Experimental Results

For each of the three evaluation methods, Tables 2 through 4 show the performance of the best strategies from the GA's last generation against opponents

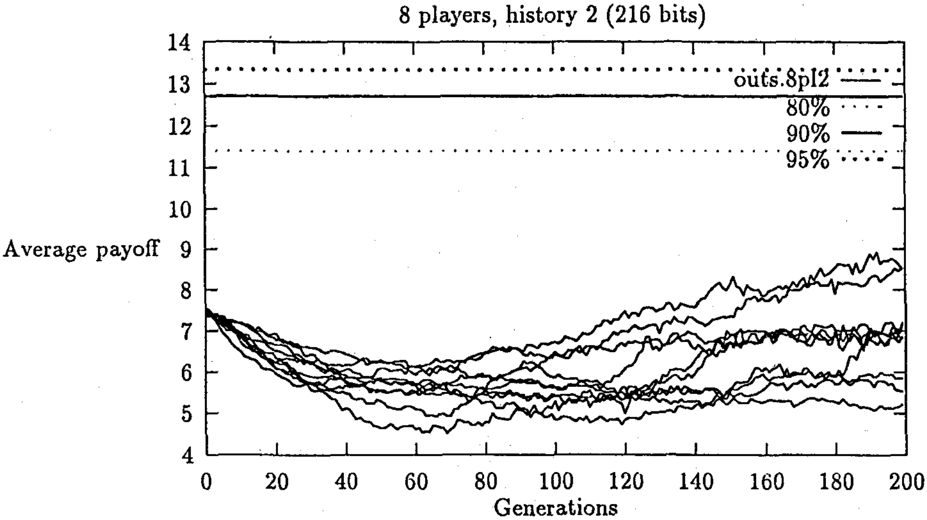

Fig. 11. For 10 runs of 8-player prisoner's dilemma with a history of 2, cooperation never emerges. The horizontal lines at the top show the 95%, 90%, and 80% levels of cooperation. To demonstrate that these runs have converged, figure 12 shows the bias of the populations.

Mean	Std Dev	Decimal	Binary genotype
8.100	0.083	1026040	1111 1010 0111 1111 1000
8.093	0.083	1022965	1111 1001 1011 1111 0101
8.091	0.083	1018871	1111 1000 1011 1111 0111
8.088	0.083	1032181	1111 1011 1111 1111 0101
8.088	0.083	1020921	1111 1001 0011 1111 1001
8.082	0.083	1028087	1111 1010 1111 1111 0111
8.077	0.083	1023990	1111 1001 1111 1111 0110
8.076	0.083	1037305	1111 1101 0011 1111 1001
8.076	0.083	1017846	1111 1000 0111 1111 0110

Table 1. Top few strategies from a partial enumerative search for strategies that play well against a large number of random opponents. This provides unseen test opponents to test the generalisation of strategies produced by co-evolution. The first 4 bits were fixed to "1", as were the eleventh through sixteenth bits. Virtually all of the best 50 strategies started by cooperating.

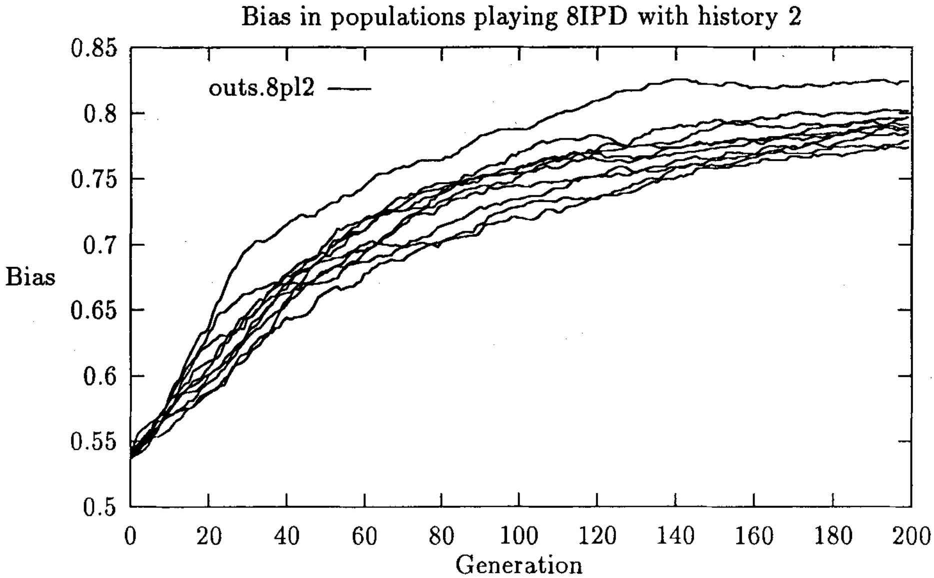

Fig. 12. In 10 runs of 8-player prisoner's dilemma with a history of 2, where cooperation never emerges, the bias demonstrates that the populations have converged. Bias is the average proportion of the most prominent value in each position. A bias of 0.75 means that, on average, each bit position has converged to either 75% "0" or 75% "1".

from (1) themselves, and (2) a pool made up of both the seen and unseen strategies from the enumerative search.

4.4 Discussion

Table 2 demonstrates that the co-evolution with the 8IPD produces strategies that are not very cooperative, as also demonstrated in Figure 11 earlier. Since the 8IPD is a game where it is easy to get exploited, co-evolution will first create strategies that can deal with non-cooperative strategies. The evolved strategies in Table 2 are cautious with each other and are not exploited by the unseen strategies from the enumerative search.

Adding fixed but not very cooperative strategies to the GA's evaluation procedure has a surprising effect. The evolved strategies in Tables 3 and 4 can cooperate well with other cooperators without being exploited by the strategies from the enumerative search, half of which it has never seen before. That is, normal co-evolution produces strategies which don't cooperate well with each other, and are not exploited by unseen non-cooperative strategies. Co-evolution with the addition of extra non-cooperative strategies gives more general strategies that do cooperate well with each other, but are still not exploited by unseen non-cooperative strategies. The experimental results also seem to indicate that the evolved strategies learn to cooperate with other cooperators better while maintaining their ability in dealing with non-cooperative strategies when the evolutionary environment contains a higher proportion of extra fixed strategies.

5 Conclusion

This paper describes two sets of experiments on the NIPD. The first set of experiments on the group size of the NIPD demonstrate that cooperation can still be evolved in the n-player IPD game where $n > 2$. However, it is more difficult to evolve cooperation as the group size increases. There are two research issues here which are worth pursuing; one is the upper limit of the group size over which cooperation cannot be evolved, the other is the quantitative relation between the group size and the time used to evolve cooperation. Glance and Huberman [5, 6] have addressed these two issues, but did not give a complete answer.

The second set of experiments in this paper deals with an important issue in co-evolutionary learning — the generalisation issue. Although the issue is the main theme in machine learning, very few people in the evolutionary computation community seem to be interested in it or address the issue explicitly and directly. We have presented some experimental results which show the importance of the environments in which each individual is evaluated, and their effects on generalisation ability.

References

1. A. M. Colman, *Game Theory and Experimental Games*, Pergamon Press, Oxford, England, 1982.
2. A. Rapoport, Optimal policies for the prisoner's dilemma, Technical Report 50, The Psychometric Lab., Univ. of North Carolina, Chapel Hill, NC, USA, July 1966.
3. G. Hardin, The tragedy of the commons, *Science*, 162:1243–1248, 1968.
4. J. H. Davis, P. R. Laughlin, and S. S. Komorita, The social psychology of small groups, *Annual Review of Psychology*, 27:501–542, 1976.
5. N. S. Glance and B. A. Huberman, The outbreak of cooperation, *Journal of Mathematical Sociology*, 17(4):281–302, 1993.
6. N. S. Glance and B. A. Huberman, The dynamics of social dilemmas, *Scientific American*, pages 58–63, March 1994.
7. R. Axelrod, The evolution of strategies in the iterated prisoner's dilemma, In L. Davis, editor, *Genetic Algorithms and Simulated Annealing*, chapter 3, pages 32–41. Morgan Kaufmann, San Mateo, CA, 1987.
8. D. M. Chess, Simulating the evolution of behaviors: the iterated prisoners' dilemma problem, *Complex Systems*, 2:663–670, 1988.
9. K. Lindgren, Evolutionary phenomena in simple dynamics, In C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, editors, *Artificial Life II: SFI Studies in the Sciences of Complexity, Vol. X*, pages 295–312, Reading, MA, 1991. Addison-Wesley.

10. D. B. Fogel, The evolution of intelligent decision making in gaming, *Cybernetics and Systems: An International Journal*, 22:223–236, 1991.
11. D. B. Fogel, Evolving behaviors in the iterated prisoner's dilemma, *Evolutionary Computation*, 1(1):77–97, 1993.
12. P. J. Darwen and X. Yao, On evolving robust strategies for iterated prisoner's dilemma, In X. Yao, editor, *Proc. of the AI'93 Workshop on Evolutionary Computation*, pages 49–63, Canberra, Australia, November 1993. University College, UNSW, Australian Defence Force Academy.
13. W. Daniel Hillis, Co-evolving parasites improve simulated evolution as an optimization procedure, In *Santa Fe Institute Studies in the Sciences of Complexity, Volume 10*, pages 313–323. Addison-Wesley, 1991.

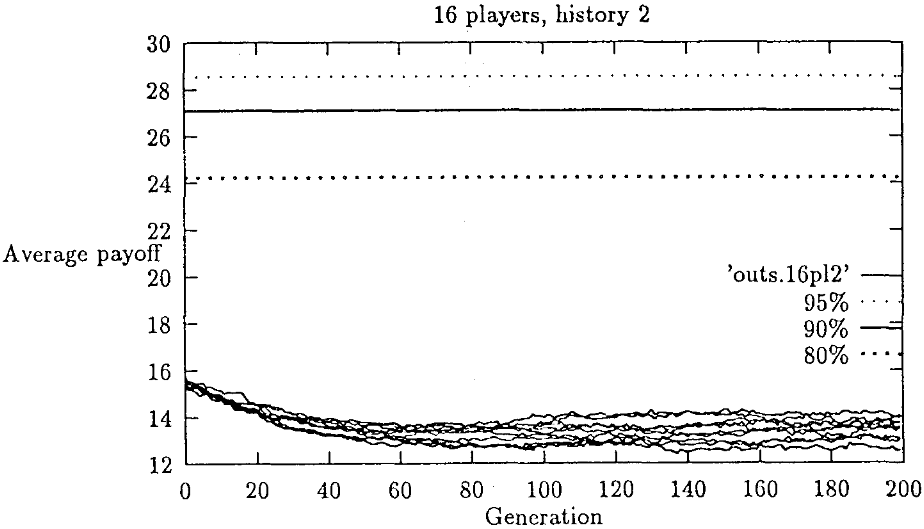

Fig. 13. For 10 runs of 16-player prisoner's dilemma with a history of 2, cooperation never emerges. The horizontal lines at the top show the 95%, 90%, and 80% levels of cooperation.

Normal co-evolution, no extra strategies in evaluation.
GA strategies play against themselves.

8pl1 (35% cooperative) against itself

	Mean	Stdv	Stdv of mean	Mean of opponents
0 11100001111011010011	7.240	3.978	0.126	6.296
1 11000001111011011011	7.285	3.985	0.126	6.392
2 11100000111111110011	7.335	3.052	0.097	8.434
3 01100010111111010011	7.258	3.202	0.101	7.889
4 11100101111111110011	7.180	4.160	0.132	5.883
5 01100000111111010011	7.000	3.090	0.098	8.111
6 11100101111111010011	7.171	4.125	0.130	6.127
7 01100001111111010011	7.241	4.027	0.127	6.286
8 00100000111111010110	7.165	3.122	0.099	8.341
9 01100100111111110110	7.706	3.523	0.111	7.949
10 10100000111111110001	7.274	3.083	0.097	8.395

GA strategies play against unseen strategies from enumerative search.

	Mean	Stdv	Stdv of mean	Mean of opponents
0 11100001111011010011	5.525	2.330	0.074	5.340
1 11000001111011011011	5.627	2.421	0.077	5.502
2 11100000111111110011	5.605	2.568	0.081	5.027
3 01100010111111010011	5.087	2.064	0.065	5.419
4 11100101111111110011	5.283	2.210	0.070	4.532
5 01100000111111010011	5.477	2.547	0.081	6.337
6 11100101111111010011	5.116	1.877	0.059	4.473
7 01100001111111010011	5.392	2.370	0.075	5.378
8 00100000111111010110	5.385	2.531	0.080	6.530
9 01100100111111110110	5.146	2.271	0.072	5.237
10 10100000111111110001	5.461	2.383	0.075	4.900

Table 2. Results of ordinary co-evolution, with no extra strategies during the GA evaluation. The GA strategies manage some cooperation among themselves, and hold their own against strategies they have not seen before.

Co-evolution, with addition of 25 fixed strategies from enumerative search.
GA strategies play against themselves.

		Mean	Stdv	Stdv of mean	Mean of opponents
0	11111000011111110100	11.678	1.715	0.054	11.965
1	11111000011111110100	11.706	1.553	0.049	11.994
2	11111000011111110110	11.440	1.603	0.051	11.922
3	11111000011111111110	11.721	1.581	0.050	12.027
4	11111000111111110100	13.264	2.521	0.080	10.636
5	11111000011111111110	11.714	1.584	0.050	12.025
6	11111000011111110110	11.420	1.669	0.053	11.895
7	11111000011111110100	11.678	1.705	0.054	11.985
8	11011000001111110100	11.618	1.781	0.056	11.958
9	11111000011111111111	11.670	1.688	0.053	11.974
10	11111000011111110100	11.649	1.697	0.054	11.973

GA strategies play against pool of 25 seen and 25 unseen strategies from enumerative search.

		Mean	Stdev	Stddev of mean	Mean of opponents
0	11111000011111110100	5.209	3.212	0.102	5.634
1	11111000011111110100	5.494	3.451	0.109	5.828
2	11111000011111110110	5.152	2.771	0.088	5.934
3	11111000011111111110	5.600	3.561	0.113	5.907
4	11111000111111110100	5.619	2.929	0.093	4.629
5	11111000011111111110	5.336	3.369	0.107	5.724
6	11111000011111110110	4.971	2.541	0.080	5.741
7	11111000011111110100	5.447	3.481	0.110	5.791
8	11011000001111110100	5.591	3.276	0.104	5.923
9	11111000011111111111	5.245	3.200	0.101	5.673
10	11111000011111110100	5.392	3.341	0.106	5.771

Table 3. Adding 25 fixed strategies to the evaluation procedure, along with the 100 co-evolving GA individuals, causes the GA to produce strategies that can co-operate more with each other, but are not exploited by the more non-cooperative strategies from the enumerative search.

Co-evolution, with the addition of 25 fixed strategies, which are 4 times as likely to be selected into the group of 8 players for 8IPD.

GA strategies play against themselves.

		Mean	Stdev	Stddev of mean	Mean of opponents
0	11111000011111110010	12.575	1.737	0.055	12.740
1	11111000011111110011	12.468	1.939	0.061	12.641
2	10111000011111010010	12.400	2.130	0.067	12.593
3	11111000011111111110	12.557	1.864	0.059	12.709
4	11111000011111110111	12.556	1.488	0.047	12.820
5	11111000011111010110	12.490	1.454	0.046	12.772
6	10111000011111110011	12.392	2.087	0.066	12.568
7	11111001011111111111	13.204	2.457	0.078	10.713
8	11111000011111111111	12.551	1.852	0.059	12.700
9	11111000011111110010	12.560	1.904	0.060	12.718
10	11111000011111110010	12.494	1.835	0.058	12.669

Best 25 strategies from GA search play against a pool of (1) 25 best from enumerative search, and (2) 25 unseen strategies from enumerative search. Note there is little diversity in the GA population.
GA strategies play against pool of 25 seen and 25 unseen strategies from enumerative search.

		Mean	Stdev	Stddev of mean	Mean of opponents
0	11111000011111110010	5.209	3.212	0.102	5.634
1	11111000011111110011	5.494	3.451	0.109	5.828
2	10111000011111010010	5.635	3.120	0.099	6.217
3	11111000011111111110	5.600	3.561	0.113	5.907
4	11111000011111110111	5.187	2.835	0.090	5.966
5	11111000011111010110	5.132	2.762	0.087	5.910
6	10111000011111110011	5.375	3.159	0.100	5.753
7	11111001011111111111	5.447	3.481	0.110	5.788
8	11111000011111111111	5.422	3.340	0.106	5.765
9	11111000011111110010	5.245	3.200	0.101	5.673
10	11111000011111110010	5.392	3.341	0.106	5.771

Table 4. Increasing the importance of the extra 25 fixed strategies causes the co-evolutionary GA to produce strategies that are even more cooperative among themselves, but are still not exploited by the unseen strategies of the enumerative search.