CHAPTER 9

Error-Correcting Codes for Team Coordination within a Noisy Iterated Prisoner's Dilemma Tournament

A. Rogers, R.K. Dash, S.D. Ramchurn, P. Vytelingum & N.R. Jennings

Electronics and Computer Science,
University of Southampton,
Southampton, SO17 1BJ, UK.
acr@ecs.soton.ac.uk.

1. Introduction

The mechanism by which cooperation arises within populations of selfish individuals has generated significant research within the biological, social and computer sciences. Much of this interest derives from the original research of Axelrod and Hamilton[1], and, in particular, the two computer tournaments that Axelrod organised in order to investigate successful strategies for playing the Iterated Prisoner's Dilemma (IPD)[2]. These tournaments were so significant as they demonstrated that a simple strategy based on reciprocity, namely tit-for-tat, was extremely effective in promoting and maintaining cooperation when playing against a wide range of seemingly more complex opponents.

To mark the twentieth anniversary of the publication of this work, these two computer tournaments were recently recreated (see `http://www.prisoners-dilemma.com/`) with separate events being hosted at the 2004 IEEE Congress on Evolutionary Computing (CEC'04) and the 2005 IEEE Symposium on Computational Intelligence and Games (CIG'05). To stimulate novel research, the rules of Axelrod's original tournaments were extended in two key ways. Firstly, noise was introduced, whereby the moves of each player would be mis-executed with some small probability. Secondly, and most significantly, researchers were invited to enter more than one player into the round-robin style tournament. This second extension to the original rules, prompted several researchers to enter teams of players into the tournament. This choice being motivated by the intuition that the members of such a team could, in principle, recognise and collaborate with one another in order to gain an advantage over other competing players. This proved to be the case, and teams of players performed well in both competitions. Indeed, a member of such a team, entered by the authors, won the noisy IPD tournaments held at both events.

Now, for this approach to be effective in practice, two key questions have to be addressed. Firstly, the players, who have no access to external means of communication, have to be able to recognise one another when they meet within the IPD tournament. Secondly, having achieved this recognition, the players have to adopt a strategy that increases the probability that one of their own kind wins the tournament. In this chapter, we present our work investigating these two questions. Specifically:

(1) We show how our players are able to use a pre-agreed sequence of moves, that they make at the start of each interaction, to transmit a covert signal to one another, and thus detect whether they are facing a competing player or a member of their own team.
(2) We show that by recognising and then cooperating with one another, the members of the team can act together to mutually improve their performance within the tournament. In addition, by recognising and acting preferentially toward a single member of the team, the team can further increase the probability that this member wins the overall tournament. In both cases, this can be achieved with a team that is small in comparison to the population (typically less than 15%).
(3) Given this approach, we show with an experimental IPD tournament that the performance of our team is highly dependent on the length of the pre-agreed sequence of moves. The length of this sequence determines both the cost and the effectiveness of the signalling between team members, and these factors contribute to an optimum sequence length that is independent of both the size of the team and the number of competing players within the tournament.
(4) Using the results of these experimental IPD tournaments, we show that signalling with a pre-agreed sequence of moves, within the noisy IPD tournament, is exactly analogous to the problem, studied in information theory, of communicating reliably over a noisy channel. Thus we demonstrate that we can implement error correcting codes in order to further optimise the performance of the team.

(5) Finally, we discuss how the results of these investigations guided the design of the teams that we entered into the two recent IPD competitions, and thus we follow this analysis with a discussion of the results of these competitions.

The remainder of this chapter is organised as follows: section 2 describes the Iterated Prisoner's Dilemma setting and related work. Section 3 describes the team players that we implemented in our investigations and section 4 describes the results of the experimental IPD tournaments that we implemented. In section 5 we analyse these results and in section 6 we discuss our use of coding theory to optimise the performance of the team. Finally, we discuss the application of these techniques within the two computer tournaments in section 7 and we conclude in section 8.

2. The Iterated Prisoner's Dilemma and Related Work

In our investigations, we consider the standard Iterated Prisoner's Dilemma (IPD) as used by Axelrod in his original computer tournaments. Thus, in each individual IPD game, two players engage in repeated rounds of the normal form Prisoner's Dilemma game, where, at each round, they must choose one of two actions: either to cooperate (C) or to defect (D). These actions are chosen simultaneously and depending on the combination of moves revealed, each player receives the payoff indicated in the game matrix shown in table 9.1. For example, should player 1 cooperate (C) whilst player 2 defects (D), then player 1 receives zero points whilst player 2 receives five points. The scores of each player in the overall IPD game are then simply the sum of the payoffs achieved in each of these rounds. In our experiments we assume that each IPD game consists of 200 such rounds, however, this number is of course unknown to the players participating.

As in the original tournaments, a large number of such players (each using a different strategy to choose its actions in each individual IPD game) are entered into a round-robin tournament. In such a tournament, each player faces every other player (including a copy of itself) in separate IPD games, and the winner of the tournament is the player whose total score, summed over each of these individual interactions, is the greatest.

Given this problem description, the goal of Axelrod's original tournaments was to find the most effective strategies that the players should adopt. Whilst in a single instance of the Prisoner's Dilemma game it is a dominant strategy for each player to defect, in the iterated game this immediate temptation is tempered by the possibility of cooperation in future rounds.

Table 9.1. Pay-off matrix of the normal form Prisoner's Dilemma game.

		Player 2	
		C	D
Player 1	C	3,3	0,5
	D	5,0	1,1

This is often termed the *shadow of the future*[14], and, thus, in order to perform well in an IPD tournament, it is preferable for a player to attempt to establish mutual cooperation with the opponent. Thus, strategies based on reciprocity have proved to be successful, and, indeed, the simplest such strategy, tit-for-tat (i.e. start by cooperating and then defect whenever the opponent defected in the last move) famously won both tournaments[2].

More recent research has extended this reciprocity based approach, and has lead to strategies that out-perform tit-for-tat in general populations. For example, Gradual[5] is an adaption of tit-for-tat that incrementally increases the severity of its retaliation to defections (i.e. the first defection is punished by a single defection, the second by two consecutive defections, and so on). Likewise, Adaptive[15] follows the same intuition as Gradual but addresses the fact that the opponent's behaviour may change over time and thus a permanent count of past defections may not be the best approach. Rather, it maintains a continually updated estimate of the opponent's behaviour, and uses this estimate to condition its future actions.

However, this reciprocity is challenged within the noisy IPD tournament. Here, there is a small possibility (typically around 1 in 10) that the moves proposed by either of the players is mis-executed. Thus a player who intended to cooperate, may defect accidentally (or vice versa)[a] and this noise makes maintaining mutual cooperation much more difficult. For example, a single accidental defection in a game where two players are using the tit-for-tat strategy, will lead to a series of mutual defections in which each player scores are reduced. This detrimental effect is often resolved by

[a]Note that this noise can be implemented in two different ways: either the cooperation is actually mis-executed as a defection, or it is simply perceived by the other player as a defection. The difference between these two implementations results in different payoffs to the players in that round on the IPD game. Whilst this does result in slightly different scores in the overall IPD tournament, it does not significantly effect the results, as, in general, the performance of a player is determined by its actions in the moves that follow either the real or perceived defection. In our experiments, we use the first implementation and assume that noisy moves are actually mis-executed.

implementing more generous strategies which do not retaliate immediately. For example, tit-for-two-tats (TFTT) will only retaliate after two successive defections[3,4] and generous tit-for-tat (GTFT) only retaliates a small percentage of the times that tit-for-tat would[4]. However, whilst these strategies manage to maintain mutual cooperation when playing against similar generous strategies, their generosity is also vulnerable to exploitation by more complex strategies. Thus effective strategies for noisy IPD tournaments must carefully balance generosity against vulnerability to exploitation, and in practise, this is difficult to achieve.

Now, the possibility of entering a team of players within a noisy IPD tournament offers an alternative to this reciprocity based approach. If the members of the team are able to recognise one another, they can unconditionally mutually cooperate and thus do not need to retaliate against defections that are the result of mis-executed moves. In addition, by defecting against players who they do not recognise as fellow team members, they are immune to exploitation from these competing players. As such, this approach resembles the notion of kin selection from the evolutionary biology literature, where individuals act altruistically toward those that they recognise as being their genetic relatives[7,8].

However, to use this approach in practise, we must address two specific issues. Firstly, we must enable the players to recognise one another and we do so by using a pre-agreed sequence of moves that each player makes at the start of each IPD interaction. Secondly, since our goal is to ensure that one member of the team wins the tournament, we explicitly identify one team member as the team leader, and have the other team members favour this individual. We describe these steps, in more detail, in the next section.

3. Team Players

Thus, as described in the previous section, we initially implement a team of players who recognise one another through the initial sequence of moves they make at the start of each IPD interaction. To this end, each team player uses a fixed length binary code word to describe this initial sequence of moves. Specifically, we denote 0 as defect and 1 as cooperate, and the binary code word indicates the fixed sequence of moves that the player should make, regardless of the actions of the opponent. This binary code word is known to all members of the team, and by comparing the moves of their opponents against this code word, players within the team can recognise if they are playing against another member of the team or against

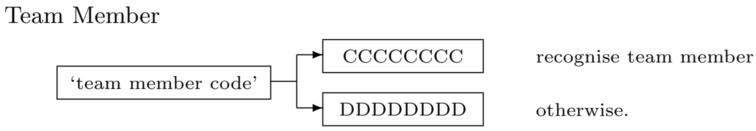

Fig. 9.1. Diagram showing the sequence of actions played by each of the team members.

an unknown opponent[b].

Now, whenever a team member meets another team member within the IPD tournament, they can recognise one another and then cooperate with one another unconditionally. In addition, the team members can recognise when they are playing against a competing player and then defect continually (see figure 9.1). In this way, since the team players no longer have to reciprocate any mis-executed moves in order to maintain cooperation, they achieve close to the maximum possible score whenever they play against other team members. In addition, since they defect against competing players, they are also immune to exploitation from these players. Thus given a sufficient number of team members within the IPD tournament, the team players perform well, compared to reciprocity based strategies.

However, our goal is to form a team that maximises the probability that one of its members will be the most successful player within the IPD tournament. Thus, we can improve the performance of the team by identifying one of the team members as the team leader, and allowing the other ordinary team members to act preferentially towards this team leader. Thus, when the ordinary team members encounter the team leader, they continually cooperate, whilst allowing the team leader to exploit them by continually defecting. In this way, whilst competing players derive the minimum possible score in interactions with the ordinary team members, the team leader derives the maximum possible score in these same interactions. Hence, by allowing the team leader to exploit them, the ordinary team members sacrifice their own chance of winning the tournament, but by changing the tournament environment, they are able to increase the chance that the team leader will win[c].

[b] Note that this recognition will not be perfectly reliable; the code word may be corrupted by noise or competing players may accidentally make a sequence of moves that matches the team code word. These are effects that we explicitly consider in section 6.

[c] Thus the team that we implement is similar to the 'master' and 'slave' approach suggested by Delahaye and Mathieu[6]. However, unlike this example, where the slaves were

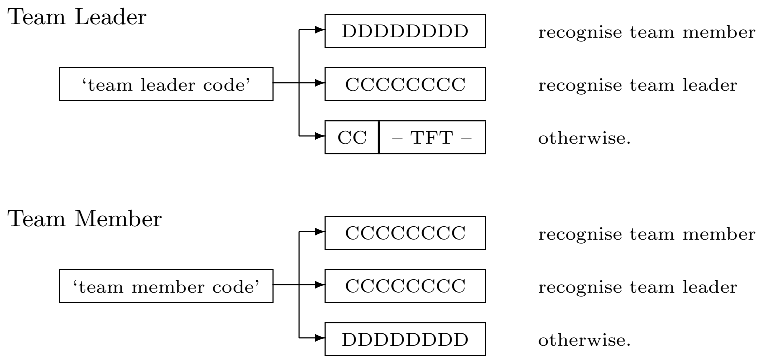

Fig. 9.2. Diagram showing the sequence of actions played by each of the team players.

The case above describes the instances in which the team leader encounters another team member. However, when the team leader encounters any other competing players it should adopt some default strategy. Clearly, using the best performing strategy available will increase the chances of the team leader winning the tournament. However, since our purpose here is to demonstrate the factors that influence the effectiveness of the team, rather than to optimise a single example case, in the investigations that we present here, we use tit-for-tat as this default strategy. As such, tit-for-tat is well understood, and whilst it does not exploit other strategies as effectively as the more recently developed alternatives discussed in the previous section, it is immune to being exploited itself. Thus in the case that the team leader does not recognise another team player, it cooperates on the next two moves in an attempt to reestablish cooperation and then continues by playing tit-for-tat for the rest of the interaction.

Finally, since the rules of the IPD tournament mean that each player must play against a copy of themselves, we also enable the team leader to recognise and cooperate with a copy of itself. Thus, the actions of both the ordinary team members and the team leader are shown schematically in figure 9.2. Note, that it is not strictly necessary to implement two different codes (i.e. one for the team leader and one for ordinary team members), however, we do so to reduce the chances of a competing player exploiting the ordinary team members (see section 7 for a more detailed discussion).

simple strategies that could potentially be exploited by any member of the population, all of our team players explicitly recognise one another and condition their actions on this recognition.

4. Experimental Results

Now, given the team players described in the previous section, two immediate questions are posed: (i) how does the number of team players within the population effect the probability that the team leader does in fact win the tournament? and (ii) how does the length of the code word (i.e. the length of the initial sequence of moves that the team players use to signal to one another) affect the performance of the team leader? In order to address these questions and to test the effectiveness of the team, we implement an IPD tournament (with and without noise) using a representative population of competing players. To ensure consistency between different comparisons within the literature, we adopt the same test population as previous researchers[5,11,15] and thus the population consists of eighteen players implementing the base strategies used in the original Axelrod competition (e.g. All C, All D, Random and Negative), simple strategies that play periodic moves (e.g. periodic CD, CCD and DDC) and state-of-the-art strategies that have been shown to outperform these simple strategies (e.g. Adaptive, Forgiving and Gradual). A full list and description of the strategies adopted by these players is provided in Appendix A.

We first run this tournament, using this fixed competing population, whilst varying the number of team players within the population, from 2 to 5 (i.e. one team leader and 1 to 4 ordinary team members), and varying the length of code word, L, from 1 to 16 bits. To ensure representative results, we also average over all possible code words, and in total, we run the tournament 1000 times and average the results. Since our aim is to show the benefit that the team has yielded, compared to the the default strategy of the team leader (in this case tit-for-tat), we divide the total score of the team leader by the total score of the player adopting the simple tit-for-tat strategy. Thus, we calculate $\langle Score_{Leader} \rangle \, / \, \langle Score_{TFT} \rangle$ and note that the greater this value, the better the performance of the team. The results of these experiments are shown in figure 9.3 for the noise free IPD tournament and in figure 9.5 for the noisy IPD tournament. In these figures, the experimental results are plotted with error bars, along with a continuous best fit curve (see section 5 for a discussion of the calculation of this line).

Now, in order to investigate the effect of larger population sizes, we also run experiments where we fix the number of team players within the population to be five (again composed of one team leader and four ordinary team members), but then generate competing populations of different sizes by randomly selecting players from our pool of 18 base strate-

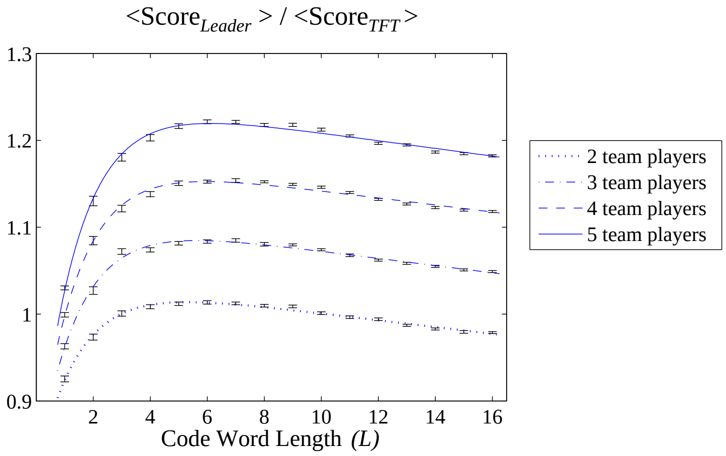

Fig. 9.3. Experimental results showing the benefit of the team in a noise free IPD tournament. Results show code word lengths from 1 to 16 bits where the total population consists of 2 to 5 team players (i.e. one team leader and 1 to 4 ordinary team members) and 18 competing players. Results are averaged over 1000 tournament runs.

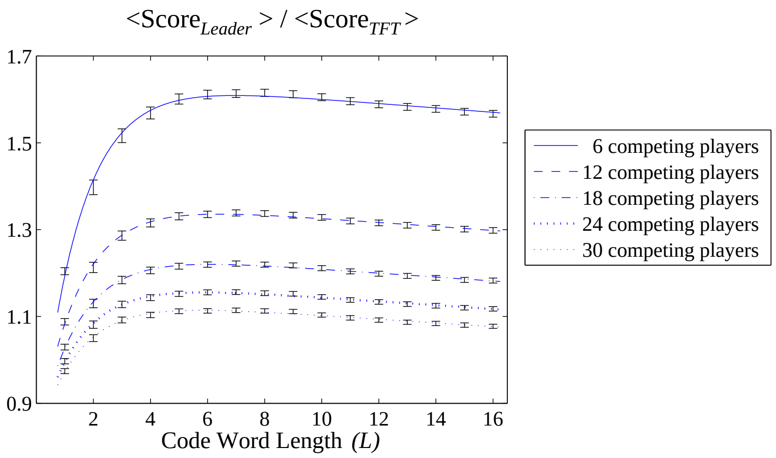

Fig. 9.4. Experimental results showing the benefit of the team in a noise free IPD tournament. Results show code word lengths from 1 to 16 bits where the total population consists of 5 team players (i.e. one team leader and 4 ordinary team members) and 6, 12, 18, 24 and 30 competing players. Results are averaged over 10000 tournament runs.

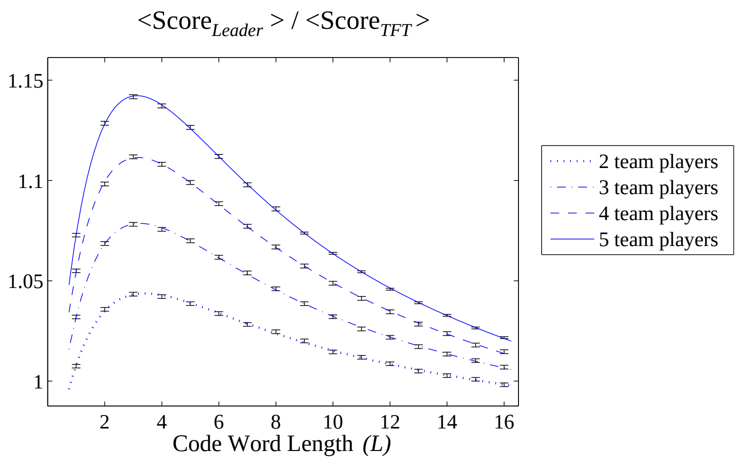

Fig. 9.5. Experimental results showing the benefit of the team in a noisy IPD tournament. Results show code word lengths from 1 to 16 bits where the total population consists of 2 to 5 team players (i.e. one team leader and 1 to 4 ordinary team members) and 18 competing players. Results are averaged over 1000 tournament runs.

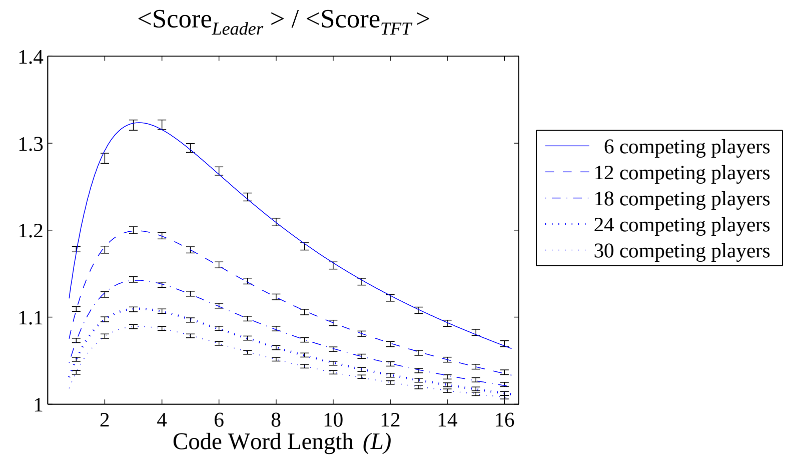

Fig. 9.6. Experimental results showing the benefit of the team in a noisy IPD tournament. Results show code word lengths from 1 to 16 bits where the total population consists of 5 team players (i.e. one team leader and 4 ordinary team members) and 6, 12, 18, 24 and 30 competing players. Results are averaged over 10000 tournament runs.

gies (always ensuring that we have at least one player using the tit-for-tat strategy). We run the tournament 10000 (more than before as we must also average over the stochastic competing population) and again calculate $\langle Score_{Leader} \rangle \,/\, \langle Score_{TFT} \rangle$. Figure 9.4 shows these results for the noise free IPD tournament and figure 9.6 show results for the noisy IPD tournament

The results clearly indicate that, as expected, increasing the number of team players, or more exactly, increasing the percentage of the population represented by the team, improves the performance of the team (i.e. increases $\langle Score_{Leader} \rangle \,/\, \langle Score_{TFT} \rangle$). In addition, in both the noise free and noisy IPD tournaments there is clearly an optimum code word length whereby the benefit of the team decreases when the code word length is longer or shorter than this optimum. Most significantly, this optimum code word length is clearly independent of both the size of the team and the population. In addition, in the case of the noisy IPD tournament, the results are very sensitive to this optimum code word length and, overall, the benefit of the team is much less than that achieved in the noise free IPD tournament. In the next section, we analyse these results and propose error correcting codes to improve performance in the noisy IPD tournament.

5. Analysis

The optimum code word lengths observed in the previous experimental results are the result of a number of opposing factors. If we initially consider the noise free IPD tournament, we can identify two such factors. The first represents the cost of the signalling between team players. As the length of the code word is increased, the team players have less available remaining moves in which to manipulate the outcome of the tournament and, thus, this factor favours shorter code word lengths. However, for this signalling to be effective, the team players must be able to distinguish between competing players and other team players. If the code word becomes too short, it becomes increasingly likely that a competing player will through pure chance make the sequence of moves that correspond to either of the code words of the team players. Thus the second factor represents the effectiveness of the signalling. It has the opposite effect of the first and thus favours longer code word lengths. The balance of these two opposing factors give rise to the behaviour seen in figures 9.3 and 9.4 where we observe an optimum code length near seven bits; at greater lengths we observe an approximately linear decrease in performance, whilst at shorter lengths, we observe a more rapid decrease in performance.

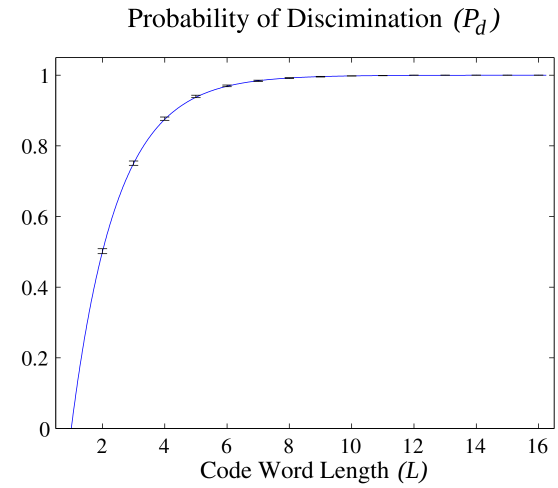

Fig. 9.7. Experimental and theoretical results showing the probability of a team player successfully discriminating between another team player and a competing player in an IPD tournament.

When noise is added to the IPD tournament, a third factor, which also affects the effectiveness of the signalling, becomes apparent. In order for the team players to recognise one another, the sequence of moves made by each player must be correctly executed. In the noisy IPD tournament, there is a small probability that one or more of the moves that constitute these code words will be mis-executed and, in this case, the team players will fail to recognise one another. The effect of this additional factor is clearly seen in a comparison of figures 9.3 and 9.4 and figures 9.5 and 9.6. In the noisy IPD tournament the optimum code word length is significantly shorter than the noise free case and there is a very rapid non-linear decrease in performance at code word lengths greater than this optimum. This final factor is very significant, and thus in the noisy IPD tournament, the team yields much less benefit than that in the noise free IPD tournament.

Now, the two factors that describe the effectiveness of the signalling can usefully be expressed as two probabilities. These are the probability that a team player will successfully discriminate a competing player from another team player, P_d, and the probability that two team players will successfully recognise one another, P_r. We can directly measure these probabilities from the experimental results presented in the last section, and then compare them to theoretical predictions.

Thus, to calculate the probability of successful discrimination, P_d, we consider that out of the 2^L possible code words, one is required for the team leader code and one for the team member code. Thus, when we consider the average over all possible code words, this probability is given by:

$$P_d = 1 - \frac{2}{2^L} \tag{1}$$

In the case of the probability of successful recognition, P_r, we require that both code word sequences are played with no mis-executed moves. If the probability of mis-executing a move is γ (in our case $\gamma = 1/10$), then this probability is simply given by:

$$P_r = (1-\gamma)^{2L} \tag{2}$$

Figures 9.7 and 9.8 show a comparison of these analytical results against the probabilities measured from the experimental results presented in the last section. Clearly the theoretical predictions match the experimental data extremely well[d] and these results indicate that the benefit of the team is strongly dependent on the effectiveness of the signalling between the team members. Most surprising, is that in the case of the noisy IPD tournament, with anything but the very shortest code word lengths, the chances of two team players successfully recognising one another is extremely small. At first sight, this result suggests that the use of teams is unlikely to be very effective in noisy environments. However, the problem that we face here (i.e. how to reliably recognise code words in the presence of mis-executed moves), is exactly analogous to that studied in information theory of communicating reliably over a noisy channel. As such, we can use the results of this field (specifically error correcting codes), to increase the probability that the team members successfully recognise one another, and thus, in turn, increase the benefit that the team will yield.

6. Error Correcting Codes

The problem of communicating reliably over a noisy channel, or in our case, reliably recognising code words when moves of the IPD game are subject to

[d]Further confirmation of this analysis is provided by the observation that the best-fit lines shown in figures 9.3 to 9.6, are calculated by postulating that the shape of the line is given by $y = A + Bx + \frac{C}{2^x} + D(1-\gamma)^{2x}$. The coefficients A, B, C and D are then found via regression so as to minimise the sum of the squared error between observed and calculated results. In the case of the noise free IPD tournament, the value of D is fixed at zero.

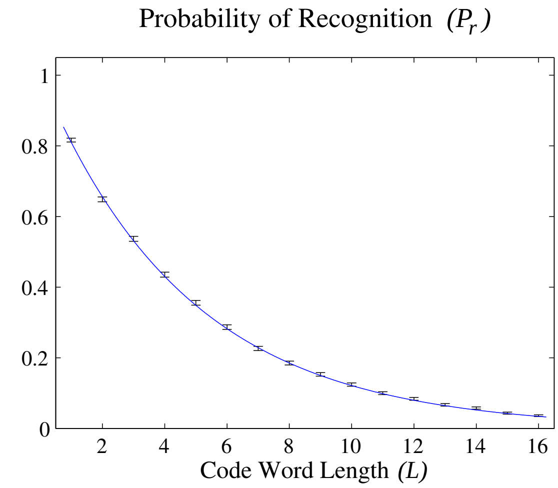

Fig. 9.8. Experimental and theoretical results showing the probability of two team players successfully recognising one another in a noisy IPD tournament.

mis-execution, is fundamental to the field of information theory[13]. One of the most widely used results of this work is the concept of error correcting codes; codes that allow random transmission errors to be detected and corrected[9,12]. Such codes typically take a binary code word of length L_c and encode it into a longer binary message of length L_m (i.e. $L_m > L_c$). Should any errors occur in the transmission of this message (e.g. a 1 transmitted by the sender is interpreted as a 0 by the receiver), the decoding procedure and the redundancy that has been incorporated into the longer message, mean that these errors can be corrected and the original code word retrieved. Different coding algorithms are distinguished by the length of the initial code word, the degree of redundancy added to the message and by the number of errors that they can correct. Thus, in our application, all the team members must implement the same coding algorithm, but now, rather than using the code word directly to describe their initial sequence of moves, they use the longer encoded message. Likewise, they observe the moves of their opponent and then compare the results of the decoding algorithm to their reference code words.

The improvement that such error-correcting codes can achieve is significant but we have several requirements when selecting an appropriate coding algorithm. The coding algorithm should increase the effectiveness of the signalling, by increasing the probability that the team members can successfully discriminate between team members and other competing players (i.e. increase P_d) and by increasing the probability that the team members recognise one another successfully (i.e. increase P_r). However, it should not increase the cost of the signalling such that this increase in effectiveness is lost. The need to limit the increase in the cost of signalling, and thus limit the length of the encoded message, L_m, is the key factor in restricting our choice of coding algorithm. As shown in figures 9.3 and 9.4, even with the perfect recognition that is achieved in the noise free case, the performance of the team begins to degrade when $L_m > 7$, and whilst many coding algorithms exist, the vast majority generate message lengths far in excess of this value[12]. Thus, our choice of coding algorithm is limited to the three presented below:

(1) A single block Hamming code that takes a 4 bit code word and generates a seven bit message that can be corrected for a single error.
(2) A two block Hamming code that simply concatenates two four bit words and thus produces a fourteen bit message that can be corrected for a single error in each 7 bit block.
(3) A [15,5] Bose-Chaudhuri-Hochquenghem (BCH) code that encodes a five bit code word into a fifteen bit message, but is capable of correcting up to three errors.

Now, in each case, the probability of successfully discriminating between team players and competing players is still determined by the initial code word length (i.e. the decoding algorithm maps the 2^{L_m} possible encoded messages onto 2^{L_c} possible code words), and thus, as before, is given by:

$$P_d = 1 - \frac{2}{2^{L_c}} \tag{3}$$

However, the probability that the team players successfully recognise one another is determined by the message length and by the error correcting ability of the code. Thus, for the Hamming code with n blocks, this probability is given by the probability that less than two error occurs in each seven bit encoded message:

$$P_r = \left[\sum_{k=0}^{1} \binom{k}{7} \gamma^k (1-\gamma)^{7-k}\right]^{2n} \tag{4}$$

For the [15,5] BCH code, the probability of recognition is given by considering that the code word can be correctly decoded if less than four errors

Table 9.2. Calculated results for the probability of discrimination, P_d, and the probability of recognition, P_r, for three different error correcting codes considered.

	Direct	**Hamming**		**BCH**
	L=3	1 block	2 blocks	[15,5]
$\boldsymbol{L_c}$ – Code Word Length	3	4	8	5
$\boldsymbol{L_m}$ – Message length	3	7	14	15
$\boldsymbol{P_d}$ – Probability of Discrimination	0.750	0.875	0.992	0.937
$\boldsymbol{P_r}$ – Probability of Recognition	0.531	0.723	0.527	0.892

occur in the fifteen bit encoded message, and thus:

$$P_r = \left[\sum_{k=0}^{3} \binom{k}{15} \gamma^k (1-\gamma)^{15-k} \right]^2 \tag{5}$$

These calculated values are shown in table 9.2 for the three coding algorithms considered, along with the original case results in which the direct code words are used (we use the value of $L = 3$ which was shown to be optimal for the noisy IPD tournament presented in section 4). Note, that all of the coding algorithms result in improvements in P_d since they all implement a code word of length greater than three. However, only the single block Hamming code and the [15,5] BCH code improve upon P_r. In the case of the two block Hamming code, the error correcting ability is not sufficient to overcome the long message length that results. Of the three algorithms, the [15,5] BCH code is superior; it creates the longest message length, yet its error correcting ability is such that it also displays the best probability of recognition. This result is confirmed by implementing the different coding algorithms within the team players and repeating the experimental noisy IPD tournament, with a fixed competing population, described in section 4. As before, to ensure representative results, we run the tournament 1000 times and average over all possible choices of code words. Table 9.3 shows the results of this comparison when 2 to 5 team players (i.e. one team leader and 1 to 4 ordinary team members) are included within the population. As expected, the [15,5] BCH code outperforms the others and, in the case where there are five team members, the performance of the [15,5] BCH algorithm is very close to the best achieved in the noise free IPD tournament presented in figure 9.3.

Finally, we present results from implementing this [15,5] BCH code in the noisy IPD tournament, again with a fixed competing population.

Table 9.3. Experimental results for $\langle Score_{\ Leader} \rangle$ / $\langle Score_{\ TFT} \rangle$ for the three different error correcting codes considered here. Tournaments are averaged over 1000 runs and the standard error of the mean is ± 0.002.

		Direct	**Hamming**		**BCH**
		L=3	1 block	2 blocks	[15,5]
Number of Team Players	2	1.043	1.055	1.044	**1.062**
	3	1.079	1.101	1.083	**1.120**
	4	1.112	1.145	1.121	**1.173**
	5	1.141	1.184	1.159	**1.221**

In table 9.4 we show the total scores achieved by each player when the number of team players increases from 2 to 5. To enable comparison with other populations, we normalise these scores and divide the total score achieved by each player, by the size of the population and by the number of rounds in each IPD game (in this case 200). Thus, the values shown are the ranked average pay-off received by the player in each round of the Prisoner's Dilemma game. Within this table, the competing players are denoted by the mnemonic given in Appendix A, the team leader is denoted by **LEAD** and the ordinary team members by **MEMB** .

Clearly, as more team members are added to the population, they are increasingly able to change the environment in which the team leader must interact and thus they are able to influence the outcome of the tournament in favour of the team leader. In three out of the four cases, the team leader is in fact the winner of the tournament, despite the fact that this player is based upon the tit-for-tat strategy that performs relatively poorly against this population (see the results shown in Appendix A). In addition, these results also clearly show that the mutual cooperation of the other team members, also leads them to perform well. Indeed, when the team consists of five (or more) such team members, all five occupy the top positions.

In table 9.5, rather than showing the averaged scores of the tournament players, we present the probability that one of the team players actually wins the overall noisy IPD tournament. In addition to the previous results where the probability that a move was mis-executed was 1/10, we present a range of values from 0 to 1/5. The results indicate that whilst we have assumed a noise level of 1/10 throughout the analysis, our results are not particularly sensitive to this value. Indeed, the more significant factor is the loss of performance of the competing players as the noise level increases. The table shows that with just two team members and no noise, a team

Table 9.4. Experimental results showing the results of the noisy IPD tournament when the team players implement a [15,5] BCH coding algorithm and there are increasing numbers of team players (a)...(d). The tournaments are averaged over 1000 runs and the standard error of the mean is ±0.002.

(a)

Player	Score
GRAD	2.347
LEAD	**2.344**
ADAP	2.263
SMAJ	2.256
GRIM	2.239
ALLD	2.219
MEMB	**2.219**
TFT	2.207
TFTT	2.175
FORG	2.171
GTFT	2.160
PCD	2.138
PCCD	2.136
STFT	2.124
HMAJ	2.109
RAND	2.101
PAVL	2.099
PDDC	2.072
NEG	2.049
ALLC	1.996

(b)

Player	Score
LEAD	**2.427**
GRAD	2.298
MEMB	**2.246**
MEMB	**2.246**
ADAP	2.228
SMAJ	2.221
GRIM	2.221
ALLD	2.192
TFT	2.168
TFTT	2.135
FORG	2.126
GTFT	2.114
PCD	2.091
STFT	2.090
HMAJ	2.084
PCCD	2.078
RAND	2.058
PAVL	2.047
PDDC	2.033
NEG	1.991
ALLC	1.934

(c)

Player	Score
LEAD	**2.503**
MEMB	**2.273**
MEMB	**2.272**
MEMB	**2.271**
GRAD	2.256
ADAP	2.191
SMAJ	2.186
GRIM	2.181
ALLD	2.161
TFT	2.133
TFTT	2.099
FORG	2.086
GTFT	2.068
STFT	2.061
HMAJ	2.054
PCD	2.047
PCCD	2.027
RAND	2.013
PDDC	2.005
PAVL	2.004
NEG	1.938
ALLC	1.877

(d)

Player	Score
LEAD	**2.568**
MEMB	**2.296**
MEMB	**2.294**
MEMB	**2.294**
MEMB	**2.292**
GRAD	2.218
ADAP	2.164
SMAJ	2.157
GRIM	2.156
ALLD	2.136
TFT	2.103
TFTT	2.062
FORG	2.054
STFT	2.036
GTFT	2.031
HMAJ	2.030
PCD	1.999
PCCD	1.982
RAND	1.969
PDDC	1.969
PAVL	1.966
NEG	1.886
ALLC	1.820

player will win the tournament just 3.4% of the time. However, as the noise level increases, the performance of the other players within the tournament degrades at a faster rate than that at which the effectiveness of the signalling between team members diminishes. At a noise level of 1/5 the same team members win 70.2% of the time. Indeed with 3 or 4 team members, the results are independent of the noise level within this range.

7. Competition Entry

The results of the previous sections clearly indicate that there is an advantage to be gained by entering a team of players into the noisy IPD tournament. However, when using these results to actually design the players for

Table 9.5. Experimental results showing the probability that one of the team members wins the noisy IPD tournament. Results are for different numbers of team members and a range of noise levels. Results are averaged over 1000 tournament runs and the standard error of the mean for each result is ±0.5.

		Noise Level (γ)				
		0.00	0.05	0.10	0.15	0.20
Number of Team Players	2	2.8 %	10.6 %	22.4 %	30.0 %	32.6 %
	3	3.4 %	81.0 %	80.4 %	81.6 %	70.2 %
	4	97.6 %	99.0 %	96.4 %	96.6 %	97.2 %
	5	97.4 %	96.6 %	97.2 %	96.6 %	96.8 %

the IPD competition entries, a number of additional factors must be considered. Firstly, in our experimental investigations we have averaged over all possible code words to produce representative results. However, for the competition entry we must actually select two code words: one for the team members and one for the team leader. Whilst the probability of recoginising a team player is independent of the choice of code word (this is a property of the codes that are implemented), the probability of succesfully discriminating between team and competing players is not. Clearly, code words that are close (in Hamming distance) to the initial moves of competing players are more likely to be corrupted by noise and thus falsely recognised. Thus we must select code words that are most unlike the moves that we expect to observe from competing players. Actually making this choice is complicated by the fact that we do not know the strategies that the competing players will use, and the moves that they make will themselves depend on the actual code words that the team players use. Thus, we again use our test population of eighteen default strategies, and by exhaustive test, we select two code words which most often lead to the correct recognition of team players and the correct discrimination of competing players.

Secondly, throughout these investigations, we have not considered the possibility of another competing player learning the code words of the team members and then attempting to exploit them. Within our competition entries, we greatly reduce the possibility of this occurring by having each team player monitor the behaviour of their opponent, in order to check that they behave as expected. Thus, if an ordinary team member recognises their opponent to be another ordinary team member, they check that the opponent does in fact cooperate in the subsequent rounds of the game. Should the opponent attempt to defect (with some allowance for the possibility of mis-executed moves), it is assumed that the opponent has been falsely recognised and thus the team member begins to defect to avoid the possibility of being exploited. Given this additional checking, the only possibility of exploitation is that a competing player learns the code word of the team leader, and thus tricks the ordinary team members into allowing themselves to be exploited. However, in the IPD tournament, this is extremely unlikely to occur. The players within the tournament only interact with each other once, thus, whilst a competing player may encounter several ordinary team members, there is little possibility of them learning the code word of the team leader in this single interaction. This is the reason for implementing separate team member and team leader code words.

Finally, we must decide how many team members to submit into the competition. Clearly, our results indicate that the larger the number of players, the better the performance of the team leader. However, typically, this number is limited by the rules of the competition (e.g. the rules of the second IPD tournament capped this number at 20), and thus, we should submit the maximum allowable number of players.

Thus, the teams that we entered into the two recent IPD competitions held at the 2004 IEEE Congress on Evolutionary Computing (CEC'04) and the 2005 IEEE Symposium on Computational Intelligence and Games (CIG'05), followed these guidelines and were successful. In the first competition, we entered several teams, that used the single block Hamming code, and a range of default strategies for the team leader. Whilst a few other researchers entered teams of players, the policy was not widely adopted and the team leader from the largest team won with a clear advantage.

In the second round of competitions we entered a single team using the more complex [15,5] BCH coding scheme, and, as in our investigations here, we used tit-for-tat as the default strategy of the team leader. In this competition, separate noise free and noisy IPD tournaments were held, and these tournaments were more competitive, as given the results of the first competition, many more researchers adopted the policy of submitting a team of players. Within the noise free IPD tournament, three of the top four positions were occupied by representatives of different teams. However, within the noisy IPD tournament, our team leader again won with a clear advantage, despite using the tit-for-tat as a default strategy. The other teams entered into this tournament performed poorly compared to the noise free IPD tournament. Thus, these results clearly illustrate the advantage that the use of error-correcting codes has yielded by enabling our team players to recognise one another in the noisy environment.

8. Conclusions

In this chapter, we presented our investigations into the use of a team of players within an Iterated Prisoner's Dilemma tournament. We have shown that if the team players are capable of recognising one another, they can condition their actions to increase the probability that one of their members wins the tournament. Since, outside means of communication are not available to these players, we have shown that they are able to make use of a covert channel (specifically, a pre-agreed sequence of moves that they make at the start of each interaction) to signal to one another and thus perform this recognition. By carefully considering both the cost and effectiveness of the signalling, we have shown that we can use error correcting codes to optimise the performance of the team and that this coding allows the teams to be extremely effective in the noisy IPD tournament; a noisy environment which initially appears to preclude their use.

Our future work in this area concerns the use of these team players in an evolutionary model of the IPD tournament. That is, rather than the static IPD tournament presented here (where the population of competing players is fixed), we consider a model where the population of competing players evolves over time (i.e. the survival of any individual within the population is dependent on their performance within an IPD tournament held at each generation). Here we are particularly interested in searching for evolutionary stable strategies (ESS), and thus are interested whether an explicit team leader is required (or indeed, can even be implemented) and how team players may attempt to exploit other team players to their own advantage. As such, this work attempts to compare the roles of kin selection and reciprocity for maintaining cooperation in noisy environments.

References

1. R. Axelrod and W. D. Hamilton. The evolution of cooperation. *Science*, 211:1390–1396, 1981.
2. R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984.
3. R. Axelrod. *The Complexity of Cooperation*. Princeton University Press, 1997.
4. R. Axelrod and J. Wu. How to cope with noise in the iterated prisoner's dilemma. *Journal of Conflict Resolution*, 39(1):183–189, 1995.
5. B. Beaufils, J. P. Delahaye and P. Mathieu. Our meeting with gradual: A good strategy for the iterated prisoner's dilemma. In *Proceedings of the Fifth International Workshop on the Synthesis and Simulation of Living Systems*, pages 202–212. MIT Press, 1997.

6. J. P. Delahaye and P. Mathieu. L'altruisme perfectionné. *Pour la Science*, 187:102–107, 1993.
7. W. D. Hamilton. The evolution of altruistic behaviour. *Am. Nat.*, 97:354–356, 1963.
8. W. D. Hamilton. The genetical evolution of social behaviour. *J. Theor. Biol.*, 7:1–16, 1964.
9. D. J. C. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.
10. M. Nowak and K. Sigmund. A strategy of win-stay, lose-shift that outperforms tit-for-tat in the prisoner's dilemma game. *Nature*, 364:56–58, 1993.
11. C. O'Riodan. A forgiving strategy for the iterated prisoner's dilemma. *Journal of Artificial Societies and Social Simulation*, 3(4):56–58, 2000.
12. W. W. Peterson and E. J. Weldon. *Error-Correcting Codes*. MIT Press, 1972.
13. C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 623–656, 1948.
14. R. Trivers. The evolution of reciprocal altruism. *Quarterly Review of Biology*, 46:35–57, 1971.
15. E. S. Tzafestas. Toward adaptive cooperative behavior. In *Proceedings of the Sixth International Conference on the Simulation of Adaptive Behavior (SAB-2000)*, volume 2, pages 334–340, 2000.

Appendix A. Test Population

The test population consists of eighteen players implementing the base strategies used in the original Axelrod competition (e.g. All C, All D, Random and Negative) plus simple strategies that play periodic moves (e.g. periodic CD, CCD and DDC) and state-of-the-art strategies that have been shown to outperform these simple strategies (e.g. Adaptive, Forgiving and Gradual). A full list and description of the strategies adopted by these players is shown in table 9.6, and table 9.7 shows the results of running noise free and noisy IPD tournaments using just these players. To ensure repeatable results, we run the tournament 1000 times and present the average results. To allow easy comparison with other publications, we normalise the scores and thus divide them by the size of the population and the number of rounds in each IPD game (in this case 200). Thus, the values shown are the ranked average pay-off received by the player in each round of the Prisoner's Dilemma game.

Note, that in this population, tit-for-tat performs relatively poorly and is easily beaten by a number of strategies. In addition, in general the scores in the noisy IPD tournament are less than those in the noise free tournament, since it is far harder to ensure mutual cooperation in the presence of accidental defections.

Table 9.6. Description of the strategies adopted by the competing players in the test population.

Strategy	**Name**	**Description**
Adaptive	ADAP	Uses a continuously updated estimate of the opponent player's propensity to defect to condition future actions[15].
All C	ALLC	Cooperates continually.
All D	ALLD	Defects continually.
Forgiving	FORG	Modified tit-for-tat strategy that attempts to reestablish mutual cooperation after a sequence of mutual defections[11].
Gradual	GRAD	Modified tit-for-tat strategy that use progressively longer sequences of defections in retaliation[5].
Grim	GRIM	Cooperates until a strategy defects against it. From that point on defects continually.
Generous Tit-For-Tat	GTFT	Like tit-for-tat but cooperates 1/3 of the times that tit-for-tat would defect[4].
Hard Majority	HMAJ	Plays the majority move of the opponent. On the first move, or when there is a tie, it cooperates.
Negative	NEG	Plays the negative of the opponents last move.
Pavlov	PAVL	Plays win-stay, lose-shift[10].
Periodic CD	PCD	Plays 'cooperate, defect' periodically.
Periodic CCD	PCCD	Plays 'cooperate, cooperate, defect' periodically.
Periodic DDC	PDDC	Plays 'defect, defect, cooperate' periodically.
Random	RAND	Cooperates and defects at random.
Suspicious Tit-For-Tat	STFT	Identical to tit-for-tat but starts by defecting.
Soft Majority	SMAJ	Plays the majority move of the opponent. On the first move, or when there is a tie, it defects.
Tit-For-Tat	TFT	Starts by cooperating and then plays the last move of the opponent.
Tit-For-Two-Tats	TFTT	Like tit-for-tat but only defects after two consecutive defections against it.

Table 9.7. Reference performance of the test population in the (a) noise free and (b) noisy IPD tournament. Results are averaged over 1000 repeated tournaments and the standard error of the mean is ±0.002.

(a)

Strategy	**Score**
ADAP	2.888
GRAD	2.860
GRIM	2.773
TFT	2.647
FORG	2.627
GTFT	2.591
SMAJ	2.575
TFTT	2.544
PAVL	2.390
ALLC	2.332
PCD	2.279
HMAJ	2.277
STFT	2.233
PCCD	2.190
ALLD	2.175
RAND	2.114
NEG	2.111
PDDC	2.081

(b)

Strategy	**Score**
GRAD	2.410
ADAP	2.329
GRIM	2.297
SMAJ	2.292
ALLD	2.278
TFT	2.245
FORG	2.211
TFTT	2.204
GTFT	2.198
PCCD	2.185
PCD	2.179
STFT	2.155
RAND	2.143
PAVL	2.140
HMAJ	2.134
NEG	2.112
PDDC	2.110
ALLC	2.043