A Reinforcement Learning Based Strategy for the Double-Game Prisoner's Dilemma*

Yang Gao

Department of Computing, Imperial College London

Abstract. The Double-Game Prisoner's Dilemma (DGPD) is a new game that incorporates the standard Prisoner's Dilemma (PD) with the Social Game. DGPD provides a novel approach in modelling economic behaviours: in each round, players not only need to choose to cooperate or defect, but also need to adjust their weights for each basic game. As a result, traditional strategies for PD, e.g. Tit-for-Tat, do not perform well in DGPD. In this paper, we propose a new strategy for the DGPD game, based on a Reinforcement Learning (RL) algorithm, namely SARSA(λ) with eligibility traces. This algorithm does not require a model of the player's environment and, therefore, is well suited for use in repeated games against unknown opponents. We empirically analyse the effectiveness of the proposed strategy. In addition, we discuss several key points in designing this strategy, e.g. the values of parameters, and show how they affect the strategy's performance. Our RL-based strategy wins the Material Versus Social Payoff Tournament (MVSPT) and significantly outperforms other strategies.

1 Introduction

Multi-Games (MGs), proposed by Edalat et al. [5], are a novel class of games used to model economic behaviours in a global economy with different markets. Consider N multi-national companies which can invest in M different countries with different cost of investment, rates of profit, labour value, interest rates, etc. In order to maximize profit, each company not only need to predict and adjust to the competitors' strategy in each country, but also need to decide in what ratio to divide its assets for investment in each country. In MGs, each player can allocate its resources in varying proportions to play in several different environments, each representing a basic game. Each player can have different sets of strategies for the different basic games. The payoff of each player in an MG is assumed to be the convex linear combination of payoffs obtained for the basic games weighted by the allocated proportions to them.

The Double-Game Prisoner's Dilemma (DGPD) is a 2-player MG consisting of two basic games: the Prisoner's Dilemma (PD) and the Social Game (SG). DGPD takes into account both the *material* payoff and the *social/moral* payoff, which are represented by the utilities of PD and the utilities of SG, respectively. The percentage of resources a player puts on the SG is represented by the *social coefficient*. DGPD provides a more comprehensive and precise model of human behaviours [5]. Details of DGPD are presented in Section 2.1.

* AT2012, 15-16 October 2012, Dubrovnik, Croatia.

However, because of the the complex nature of this game, traditional successful strategies for PD does not perform well in DGPD. For example, the traditional Tit-for-Tat (cooperate on the first move and then reciprocate the opponent's last action) performs poorly in DGPD because it does not take resource allocation (i.e. social coefficient adjusting) into account.

Reinforcement Learning (RL) is based on the idea that the tendency to produce an action should be strengthened (reinforced) if it produces favourable results, and weakened if it produces unfavourable results. SARSA(λ) [14] is a RL algorithm that does not need a model of the environment and can be integrated with other techniques, e.g., eligibility traces, reward shaping, etc., to improve its convergence speed [14]. Therefore, it is well suited for use in repeated games against an unknown opponent.

In this paper, we propose a new RL-based strategy for the DGPD game. In particular, our strategy consists of two parts: *social coefficient updating* and *decision-making*. In the first part, we update the social coefficient according to the payoff matrix of DGPD and both agents' action in the last round. Upon the updated social coefficient, the decision-making part builds a *Markov Decision Process* (MDP) and uses RL to adjust to the opponent's behaviours. Our RL-based strategy shows outstanding performances in the Material Versus Social Payoff Tournament (MVSPT)[1], a tournament to evaluate the effectiveness and robustness of each strategy in the DGPD scenario.

The remainder of this paper is organised as follows: Section 2 provides the background of the DGPD game and related theories, and introduces the structure of MVSPT. Section 3 gives details of our strategy and discusses several key issues in designing it. Section 4 compares the performance of our strategy with that of the other candidates in the tournament. Section 5 describes related works, including other sophisticated strategy for DGPD and other RL-based strategies for standard PD. At last, we conclude our research in Section 6.

2 Background

We first introduce MG and DGPD, a 2-player MG consisting of two basic games: the PD and the SG. Then we give some fundamentals of the concept of Nash Equilibrium. After that, RL will be briefly described and a specific RL algorithm: SARSA(λ) with eligibility traces, will be presented in detail. Finally, we introduce the rules and restrictions of MVSPT.

2.1 Multi-Games and Double-Game Prisoner's Dilemma

Formally, Multi-Games (MGs) are defined as follows [5]: consider M finite N-player games $G_i (1 \leq i \leq M)$ with the strategy set S_{ij} and the payoff matrix π_{ij} for player $j (1 \leq j \leq N)$ in the game G_j. Assume each player j has a set of M weights η_{ij} with $\sum_{j=1}^{M} \eta_{ij} = 1$. The N-player MG G with basic games G_i is defined as the finite

[1] Part of the documents and source code of MVSPT can be downloaded at `https://github.com/theoboyd/mvspt`

strategy game with players $j(1 \leq j \leq N)$ each having strategy set $\prod_{1 \leq i \leq M} S_{ij}$ and payoff

$$\pi_j(\prod_{1 \leq i \leq M} s_{ij}) = \sum_{1 \leq i \leq M} \eta_{ij} \pi_{ij}(s_{ij})$$

for strategy profile $s_{ij} \in S_{ij}$ and possibly incomplete information (types) η_{ij} for $1 \leq i \leq M$. We say a MG is *uniform* if for each player j, the set S_{ij} is independent of the game G_i, i.e., we can write $S_{ij} = S_j$, and $s_{ij} = s_{i'j}$ for $1 \leq i \leq i' \leq M$.

The Double-Game Prisoner's Dilemma (DGPD) is a uniform 2-player MG consisting of two basic games: PD and SG. PD is an abstraction of social situations where each agent is faced with two alternative actions: cooperating, i.e., doing the socially responsible thing, and defecting, i.e., acting according to self-interest regardless of how harmful this might be to the other player. PD is considered as a standard method for modelling social dilemmas, and has also been used to model conditional altruistic cooperation, which has been tested by real monetary payoffs [8]. The payoff matrix of PD is shown in Table 1. The payoff values in the PD should satisfy the following two inequalities:

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

$$R > (T + S)/2 \tag{2}$$

Inequality 1 specifies the order of the payoffs and defines the dilemma, since the best a player can do is to get T (temptation to defect), followed by R (reward for cooperation), P (punishment for mutual defection) and S (sucker's payoff). On the other hand, Inequality 2 ensures that in the repeated game, the players cannot get out of the dilemma by taking turns in exploiting each other.

	Player 2 Cooperate	Player 2 Defect
Player 1 Cooperate	(R, R)	(S, T)
Player 1 Defect	(T, S)	(P, P)

Table 1. Payoff matrix of PD

However, when faced with the choice of cooperate or defect, human beings not only consider their material score, but also the social and moral payoffs. According to some research [17, 18], human social evolution has a moral direction which has extended our moral campus in the course of thousands of years of increasing communication and complexity in human social organisation. The same applies to economic decisions made by corporations or governments [5], in which actions taken can have significant social and environmental implications, which are not incorporated in the material gains they produce. A more adequate model of human behaviour, therefore, should take into account these aspects of social evolution as well. As a result, SG has been proposed and integrated into PD [4]. SG encourages cooperation and discourages defection, as

cooperating is usually considered to be the ethical and moral choice to make when interacting with others in social dilemmas. The payoff matrix of SG is shown in Table 2.

	Player 2 Cooperate	Player 2 Defect
Player 1 Cooperate	(M_1, M_2)	(M_1, M_2')
Player 1 Defect	(M_1', M_2)	(M_1', M_2')

Table 2. Payoff matrix of SG

The payoff of DGPD for each player are then the weighted sum of the payoffs of PD and SG, represented by $MaterialScore$ and $SocialScore$ respectively, using the player's *social coefficient*, which reflects how much they cares about the morality of their actions. The payoff matrix of DGPD is presented in Table 3, where η and γ (with $0 \leq \eta, \gamma \leq 1$) are social coefficients of Player 1 and Player 2, respectively.

	Player 2 Cooperate	Player 2 Defect
Player 1 Cooperate	$(1-\eta)R + \eta M_1, (1-\gamma)R + \gamma M_2$	$(1-\eta)R + \eta M_1, (1-\gamma)T + \gamma M_2'$
Player 1 Defect	$(1-\eta)T + \eta M_1', (1-\gamma)S + \gamma M_2$	$(1-\eta)P + \eta M_1', (1-\gamma)P + \gamma M_2'$

Table 3. Payoff matrix of DGPD

With respect to the payoff values, the following formulae hold:

$$T > R > M_1 = M_2 > P > S \tag{3}$$

$$M_1 = M_2 > (R + P)/2 \tag{4}$$

$$M_1' = M_2' = S \tag{5}$$

Inequality 3 ensures that the social payoffs, namely M_1 and M_2, are neither over-evaluated ($T > R > M_1 = M_2$), nor under-evaluated ($M_1 = M_2 > P > S$), so as to maintain the dilemma and encourage social behaviours at the same time. Inequality 4 has the similar function as Inequality 2. Equality 5 indicates that M_1' and M_2' should be equal to S, in order to discourage defection with a high social coefficient, which would be self-contradictory, and to punish defection, since M_1' and M_2' are the payoff value for defection in the SG, which, by its definition, should not give high value to defection.

2.2 Nash Equilibrium

The notion of Nash Equilibrium (NE) is a solution concept of a game involving two or more players, in which each player is assumed to know the equilibrium strategies of

the other player, and neither player has any incentive to change only its own strategy unilaterally. In particular, John Nash has proved that every game in which every player has a finite set of possible strategies has a Nash equilibrium in mixed strategies [11].

With respect to the PD game, the only NE accounts to both players defecting: since $T > R > P > S$ (see Section 2.1), according to the payoff matrix (see Table 1), each player can unilaterally improve their situation by defecting regardless of the other's decision. However, it is obvious that the score of both player cooperating is higher than that of both player defecting. So the NE solution is not necessarily the optimal solution.

2.3 Reinforcement Learning and SARSA(λ)

Among many mathematical models of RL, MDP is the most widely used one and has several variants [14]. An MDP is a tuple (S, A, T, R), where S is the state space, A is the action space, $T(s, a, s') = Pr(s'|s, a)$ is the probability of moving from state s to state s' by executing action a, and $R(s, a, s')$ gives the immediate reward r received when action a is taken in state s, moving to state s'.

However, in most real environments, the transition probabilities and reward functions, namely the model of the environment, are not known. In these cases, *on-policy* learning algorithms are used, which apply temporal-difference updates to propagate information about values of states, $V(s)$, or state-action pairs, $Q(s, a)$. These updates are based on the difference of the two temporally different estimates of a particular state or state-action value. The SARSA(λ) [14] algorithm is such an on-policy learning algorithm. In addition, we integrate SARSA(λ) algorithm with replacing eligibility traces, which keep a temporary record of the occurrence of each event, to improve the convergence speed of the algorithm [14]. The details of this algorithm is presented in Algorithm 1. In the algorithm, α is a learning rate parameter and γ is a discount factor governing the weight placed on the future. The value e represents *eligibility traces*,

Algorithm 1 Learning algorithm: SARSA(λ) with replacing eligibility traces (adapted from [14])

Initialise $Q(s, a)$ arbitrarily for all states s and actions a
Repeat (for each episode):
  Initialise $e(s, a) = 0$ for all s and a
  Initialise current state s_t
  Choose action a_t from s_t using the policy derived from Q
  Repeat until s_t is the terminal state:
    Execute action a_t, observe reward r_t and new state s_{t+1}
    Choose a_{t+1} from s_{t+1} using the policy derived from Q
    $\delta \leftarrow r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)$
    $e(s_t, a_t) \leftarrow 1$
    For all s, a:
      $Q(s, a) \leftarrow Q(s, a) + \alpha\delta e(s, a)$
      $e(s, a) \leftarrow \gamma\lambda e(s, a)$
    $s_t \leftarrow s_{t+1}$; $a_t \leftarrow a_{t+1}$

which stores the credit that previous action choices should receive for current rewards, while λ governs how much credit is delivered back to them. The action selection rule used in line 5 and line 8 is ϵ-*greedy*: the action with highest $Q(s, a)$ value will be selected for a proportion $1 - \epsilon$ of the trials; for the other ϵ proportion, actions will be selected randomly.

2.4 Material Versus Social Payoff Tournament

MVSPT is based on the DGPD game. The structure of the tournament is *round-robin*, i.e., all the strategies compete in the iterated double game against all the other strategies and themselves once. Each game between any two strategies consists of a variable number of rounds. The number of rounds will in the range of 100 to 200, but the exact number will not be given to each strategy to prevent the degenerate case of total mutual defection[2]. The winner of the tournament is the strategy which has the highest *total score*:

$$((1 - \beta) \times MaterialScore) + (\beta \times SocialScore)$$

where β is the *global social coefficient* that represents the social inclination, and hence weighting and importance of the social score to society as a whole. In order to encourage both material and prosocial play, in MVSPT β will be fixed to 0.5.

However, instead of just showing the total score, the social and material scores will be kept separate. As strategies play further rounds, their material and social scores are tallied separately and recorded as two running totals that, along with η, completely represent the current state of the player. We believe this is a better way of formulating the dual game as it does not lose the meaning of the two scores by themselves and allows us (and strategies) to measure social contribution separately and with equal importance to material contribution.

The numerical values of the payoffs used in this tournament are:

$$T = 5, R = 3, M_1 = M_2 = 2.5, P = 1, M_1' = M_2' = S = 0$$

As described in Section 2.1, these values are chosen precisely to create an environment where a real dilemma is present. There are some other restrictions for this tournament:

- In order to simplify the system, each agent's private social coefficient (η) is a discrete value. It is fixed to one of these six values: $\{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$.
- To model human dynamics in social behaviour more accurately, certain restrictions are imposed on how much a player can increase or decrease the social coefficient in a single round. In particular, an agent can only increase or decrease their social coefficient by 0.2 in each round, since, in general, humans do not change their moral values radically in a short amount of time [5].

[2] If the total number of rounds, n, was known, a strategy would plan to defect at round n as there is no repercussion for doing so and they can guarantee the maximum payoff. Similarly, they can also defect in round $n - 1$ because they and their opponent will have (theoretically at least) defected in round n. This continues backwards until defection is planned in all rounds.

– In each round (except the first round), a player is able to read his own as well as the opponent's action in the last round. However, each agent's social coefficient (η) is always private to itself[3].
– It is not permitted to attempt to cooperate just after having defected and reduced one's social coefficient. Similarly, it is not permitted to attempt to defect just after having cooperated and raised one's social coefficient. Both of these measures are in place to prevent undermining the concept of the game and if a strategy attempts to perform one of these combinations, it is invalid.

3 Our Approach

The major difference between DGPD and standard PD is that in DGPD, agents not only need to choose to defect or cooperate in each round — as they do in the standard PD — but also need to update their weights on each game, namely the social coefficient values. In accordance with this characteristic, our RL-based strategy consists of two components: *social coefficient (η) updating* and *decision-making*. The RL process is only involved in the decision-making part. In Section 3.1, the η-updating rules will be presented and discussed, and in Section 3.2, the details of the RL algorithm will be given.

3.1 Social Coefficient (η) Updating

We design our η-updating strategy as a *fixed-history* strategy, namely η will be updated based on both players' actions in the last k rounds, where k is the *length of the history*. Same with the setting of the η-updating rules in [5], we set k as 1 here. According to the payoff matrix of DGPD (see Table 3), we analyse the updating rules in each situation as follows (each situation is represented by (M, O), where M is my last action and O is the opponent's last action; with respect to actions, C represents cooperate and D represents defect):

– (C, C): Assume that both sides still mutually cooperate in the next round. Then by decreasing η, a higher payoff can be gained. Similarly, if it is (D, C) or (D, D) in the next round, decreasing η is still the better choice. Note that η represents the weights given to the SG. Hence, to varying degrees, it reflects the agent's willingness to cooperate. As a result, by decreasing η, our strategy is very unlikely to cooperate in the next round. So even though by decreasing η the payoff of (C, D) will drop, we still believe that decreasing η is the best choice in (C, C).
– (C, D) and (D, D): In these situations, instead of attempting to alter the opponent's behaviour by retaliating to its defections, we realise that it is better to increase the η value to get the social reward from cooperation and avoid the disastrous cycles of mutual defection, which would be caused by a low η value.

[3] Because the social coefficient of the opponent, namely γ, is unknown to us, we ignore the effect of γ when we discuss the social coefficient updating rules in Section 3.1

– (D, C): In this situation, we will decrease the η value, as we see that the opponent does not retaliate to our defection. In a sense we exploit the opponent in this particular case, however, it is the opponent who allows that behaviour by not retaliating. In such cases, some strategies allow themselves to be taken advantage of, because if they have a high η value, they can get higher payoff from cooperation and not concern themselves about their opponent's actions.

3.2 RL-Based Decision-Making

As described in Section 3.1, the η values are updated according to both sides' actions in the last round, so it is reasonable to believe that the updated η value is able to compactly summarise past sensations and roughly represent the history. In other words, η satisfies the *Markov Property* and, as a result, an MDP can be built upon η. To be specific, the state set S (see Section 2.3) can be set to $\{V_\eta\}$, where V_η is the value of η. As described in Section 2.4, there are only six values available for η, so there are six elements in S. Action set $A = \{C, D\}$, where C and D represents cooperate and defect respectively. Since the opponent's action will affect the next state (η value), the transition probability $T(s, a, s')$ is unknown. Immediate reward $R(s, a, s')$ are defined as the payoff of doing action a in state s and moving to s'. Because we are using SARSA, the model of the environment is not needed [14]. As a result, if the opponent's strategy is fixed, namely the transition probability of each tuple (s, a, s') (i.e. $T(s, a, s')$) is fixed, our SARSA learning algorithm is able to find the optimal action in each state if each state-action pair (s, a) can be visited for infinite times [14].

However, as described in Section 2.4, each game consists of only 100 to 200 rounds. So each state-action pair can be visited for 200 times at most, which is far fewer than the theoretical requirement of converge (i.e. infinite times). In other words, in order to ensure that RL can find the optimal action in each state (η value), some prior knowledge about the environment is essential, so as the speed up the convergence process. As a result, the initial $Q(s, a)$ values, for all state s and action a, need to reflect some prior knowledge about the opponent.

Babes et al. [2] has suggested that by assigning $Q(s, a)$ to be the maximum total expected total discounted reward from state-action pair (s, a) can make learning optimal behaviour nearly immediate. To simplify this initialization process, we assume that the expected total discounted reward equals the immediate reward of executing action a in situation s. Therefore, each $Q(s, a)$ can be initialized as follows:

$$Q(V_\eta, C) = P_{oc} \times Payoff(V_\eta, (C, C)) + (1 - P_{oc}) \times Payoff(V_\eta, (C, D)) \quad (6)$$

$$Q(V_\eta, D) = P_{oc} \times Payoff(V_\eta, (D, C)) + (1 - P_{oc}) \times Payoff(V_\eta, (D, D)) \quad (7)$$

where $Payoff(V_\eta, (M, O))$ is the payoff when $\eta = V_\eta$ and my last action and the opponent's last action are M and O, respectively. P_{oc} is the probability of the opponent to cooperate. $Payoff(V_\eta, (M, O))$ can be calculated by using the payoff matrix shown in Table 3 (for player 1). In addition, according to a principle in designing PD strategies: *be nice* (see [1]), we always choose to cooperate in the first round of each game. The complete RL-based strategy is shown in Algorithm 2.

Algorithm 2 RL-based strategy for DGPD

Initialisation:
initialise η and P_{oc}
for all state s and action a:
initialise $Q(s, a)$ according to formula 6 and 7

Response:
if first round:
do not change η
$s_t = V_\eta$; $a_t = C$
return a_t
otherwise:
read the opponent's last action A_o
η**-updating**:
if $(a_t, A_o) = (C, C)$: decrease η
else if $(a_t, A_o) = (C, D)$: increase η
else if $(a_t, A_o) = (D, C)$: decrease η
else if $(a_t, A_o) = (D, D)$: increase η
else: do not change η
decision-making:
$reward = Payoff(s_t, a_t, A_o)$
$s_{t+1} = V_\eta$
choose action a_{t+1} from s_{t+1} using the policy derived from Q ($\epsilon - greedy$)
$\delta \leftarrow reward + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)$
$e(s_t, a_t) \leftarrow 1$
for all state s and action a:
$Q(s, a) \leftarrow Q(s, a) + \alpha\delta e(s, a)$
$e(s, a) \leftarrow \gamma\, e(s, a)$
$s_t \leftarrow s_{t+1}$; $a_t \leftarrow a_{t+1}$
return a_t

In Algorithm 2, the **Initialisation** part is executed at the beginning of each game. When a game begins (our strategy begins to play with the opponent), our strategy will behave according to the **Response** part. To be specific, we first read the opponent's last action A_o from the tournament simulation platform (see Section 2.4). In the η-updating part, by saying increase or decrease η, the value of η will be increased or decreased by 0.2 respectively (see Section 2.4). Then the updated η value as well as A_o will be served as input of the decision-making part. This part is basically same with the repeating part of the SARSA(λ) algorithm (see Algorithm 1 between Line 7 and Line 14). The strategy will terminate when a game is finished.

4 Implementation and Empirical Results

In this section, we will discuss the values of the parameters in our strategy in Section 4.1, and show the performance of our strategy in the tournament (Section 4.2).

4.1 Parameter Values

The values of each parameter shown in this paper are proposed on the basis of observation, the principles for designing PD (see [1]) and earlier results. We implement five strategies[4] in our experiments: RL-based strategy, Tit-for-Tat, AlwaysRandom, AE-GS and AE-GSS. Tit-for-Tat initialises η randomly and keeps its value constant during the whole process of each game. The AlwaysRandom strategy randomly initialises η at the beginning of each game and randomly updates its value in each round of a game. Based on the updated η value, AlwaysRandom chooses to cooperate if $\eta \geq 0.5$; otherwise, it chooses to defect. AE-GS/AE-GSS are presented in detail in Section 5. However, we cannot ensure that these strategies we implement in our experiments are exactly the same as the strategies used in the real tournament, even though they may fall under the same name. In our experiments, each game consists of 200 rounds of play, but the agents are not able to access to the exact number of rounds for the reason described in Section 2.4. It is worth to notice that as all the parameter values discussed in this section are the results of a limited number of experiments, we make no claims that they are optimal, and very likely there is considerable room for improvement.

All the parameters involved in this strategy fall into two categories: the parameters in standard RL, including α, γ, λ and ϵ; and the parameters specific to this strategy: η and P_{oc}. In terms of the parameters in standard RL, since the effect of each of them is clear (see Section 2.3) and a lot of research has been done to discuss how to choose their values (e.g., [9, 14]), we give some conventional and classical values to them: $\alpha = 0.125$, $\gamma = 1.0$, $\lambda = 0.8$ and $\epsilon = 1$. Furthermore, we make ϵ linearly decrease from 1 to 0 during the first 100 rounds in each game. By doing this, each state-action pairs can have more opportunities to be visited in the early stage, so as to get more

[4] Because we cannot access all the submitted strategies of MVSPT, we only implement five selected strategies. However, even though our strategy collection is smaller than that of MVSPT, according to the final performance (see Section 4.2 and Table 4), the top three strategies, i.e., RL-based, AE-GS and AE-GSS, are all included in our experiments. Therefore, the results derived from our experiments are still valuable.

accurate $Q(s, a)$ values for each state-action pair; moreover, since the exploration rate decreases to 0 in the 100 rounds of each game, in the remaining rounds there will be no explorations and, as a result, the performance can be stable and nearly optimal.

Given the settings above, we then analyse the impact of η and P_{oc} on the strategy's performance. We increase the value of η and P_{oc} from 0 to 1 in the steps of 0.2 and 0.1, respectively. The effect of η and P_{oc} on the strategy's performance is shown in Figure 1. The strategy's performance reaches its peak when $\eta = 0.6$ and $P_{oc} = 0.5$. When the value of η is fixed, the performance does not change significantly with the change of P_{oc}; however, when we set P_{oc} as constant and decrease the value of η from 1 to 0, the performance keeps stable when $\eta > 0.2$, but drops dramatically when $\eta \leq 0.2$. Recall that η represents the agent's weight put on social score (see Section 2.1) and P_{oc} represents the probability of the opponent to choose to cooperate (see Section 3.2). Thus, this result indicates that we should be neither overoptimistic (very high η and P_{oc} value) nor overpessimistic (very low η and P_{oc} value). In addition, since the performance is better when both η and P_{oc} are relatively large (larger than 0.5), this result also corresponds with an important principle of designing PD strategy: be nice (see [1]). To summarise, in order to ensure that the strategy performs nearly optimally and stably when competing with a large collection of strategies in MVSPT, in our version the parameters are set as: $\alpha = 0.125$, $\gamma = 1$, $\lambda = 0.8$, $\eta = 0.6$, $P_{oc} = 0.5$ and $\epsilon = 1$ in the beginning of each game and linearly decreases to 0 in the first 100 rounds.

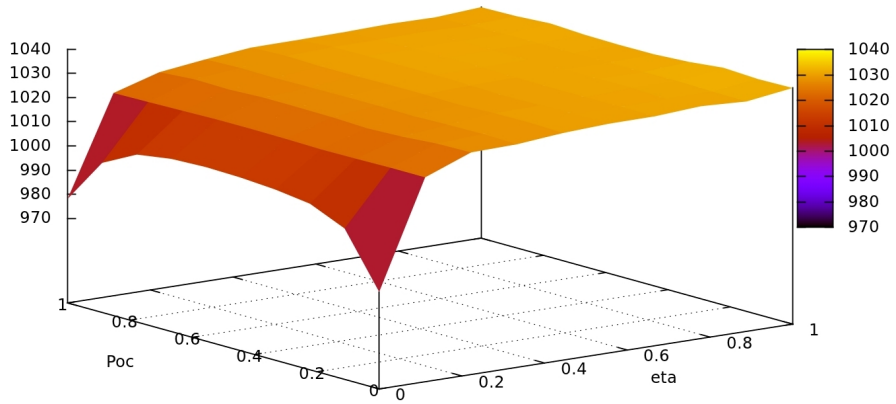

Fig. 1. The effects of η and P_{oc} on the performance of RL-based strategy. The vertical axis represents the total score the strategy gets. All the scores are averaged over 1000 independent trails.

4.2 Empirical Results

The result of MVSPT can be checked online: `http://www.doc.ic.ac.uk/~tb208/mvspt-results.php`[5]. Because we cannot access all of the top 10 strategies, we are not able to present and analyse their technical details in this paper. For more information of these strategies, the official website of MVSPT (`http://www.doc.ic.ac.uk/~tb208/mvspt-main.php`) may be referred to. The performance of the top 10 strategies have been listed in Table 4.

Strategy name	Win percentage	Average η	Material Score	Social Score
RL-based	79.91%	0.409	0.923	0.417
AE-GS-S	8.92%	0.791	0.392	0.771
Rafal Strategy	4.78%	0.773	0.438	0.754
AE-GS	3.97%	0.818	0.401	0.795
Double Increment	2.41%	0.836	0.340	0.839
A Bit Random	0.01%	0.700	0.541	0.533
Always Cooperate	0.00%	1.000	0.000	1.000
Always Defect	0.00%	0.000	1.000	0.000
Nash Defect	0.00%	0.000	1.000	0.000
Tit-for-Tat	0.00%	0.000	0.948	0.000

Table 4. The performance of top 10 strategies averaged over 10000 round-robin simulations.

In Table 4, the second column (*win percentage*) shows how much percentage a strategy's total score (see Section 2.4) is higher than that of Tit-for-Tat. The third column shows the average social coefficient value of each strategy. The higher this value is, the the strategy is more likely to cooperate. Column 4 and Column 5 present the scaled material score and social score (see 2.1), respectively. The results indicate that our RL-based strategy significantly outperforms Tit-for-Tat as well as other sophisticated strategies, e.g., AE-GS-S and AE-GS (for more details about these two strategies, see Section 5). An interesting phenomenon is that among all the top 7 strategies, the average social coefficient value and the social score of the RL-based strategy is the lowest. This result implies that under the current setting of MVSPT, a successful strategy should be more material rather than be socially beneficial. However, the principle *being nice* has also been shown as important in choosing the value of parameters (see Section 4.1). Therefore, how to keep the balance between being social and being material is the key point in designing a successful strategy for this tournament.

[5] Since this website is currently on a student's domain of Imperial College London, due to the college's policy, it may be removed or moved to other places. If the website is unavailable, please contact the author of this paper

5 Related Work

Because of the novelty of DGPD, there are few published sophisticated strategies for this game. The only exception, to the best of our knowledge, is the AE-GS-S and AE-GS strategy proposed by Edalat et al. [5]. Their strategies also consists of two parts: η-updating and decision-making. With respect to the η-updating rules, the only difference between their strategy and our strategy locates at the situation when both player cooperate in the last round, namely (C, C): in our strategy, we choose to decrease η (see Section 3.1), whereas they choose to increase η (AE-GS strategy) or do not change its value (AE-GS-S strategy). The decision-making part of AE-GS and AE-GS-S strategy are same: first they analyse the NE under different social coefficient values (both self coefficient η as well as the opponent's coefficient γ) and find the best action(s) in each situation. When compete with other strategies, they give a fixed model to predict the opponent's behaviour and social coefficient (γ). Based on its own social coefficient (η) and the prediction of γ, AE-GS/AE-GS-S find the best action in this situation. So a main difference between AE-GS/AE-GS-S and our RL-based strategy is that their strategy predict the opponent's behaviour rigidly, whereas our strategy learns the opponent's behaviour dynamically. The AE-GS/AE-GS-S strategy is presented in Algorithm 3.

Algorithm 3 AEGS/AEGSS algorithm

Initialise:
  $\eta = 0.0$
Response:
  if first round:
    do not change η; return C
  otherwise:
    read the opponent's last action A_o
    η**-updating**:
      if $(a_t, A_o) = (C, C)$: increase η (AE-GS) or do not change η (AE-GS-S)
      else if $(a_t, A_o) = (C, D)$: increase η
      else if $(a_t, A_o) = (D, C)$: decrease η
      else if $(a_t, A_o) = (D, D)$: increase η
      else: do not change η
    decision-making:
      if $\eta \leq 0.2$: return D
      else if $\eta \leq 0.4$:
        for 30% proportion: return D
        for 70% proportion: return C
      else: return C

There is some research on RL-based strategy in the Iterated PD (IPD) game. Sandholm et al. [13] studied the RL-based strategies in IPD along three dimensions: the length of history the agents received as context, the type of memory the agents employed and the exploration schedule the agents followed. They concluded that although all the learners faced difficulties when playing against other learners, agents with longer history windows, longer lookup table memories, and longer exploration schedules fared

best in the IPD games. Furthermore, Babes et al. [2] proposed *social reward shaping* to improve the learning performance in multi-agent learning scenarios. In particular, they showed that learning agents with social reward shaping are able to both lead — encourage adaptive opponents to stably cooperate — and follow — adopt a best-response strategy when paired with a fixed opponent — where other existing approaches achieve only one of these objectives. Although these research are based on IPD, they still give us some insights into how to design RL-based strategies in DGPD.

6 Conclusion

In this paper, we propose a RL-based strategy for the novel Double-Game Prisoner's Dilemma (DGPD) game. In this game, each player not only needs to choose to cooperate or defect in each game round, but also needs to give different weights to two basic games: the Prisoner's Dilemma (PD) game and the Social Game (SG). In accordance with the unique structure of DGPD, our strategy consists of two parts: social coefficient (η) updating and decision-making. The η values are updated based on the payoff matrix of DGPD as well as both players' actions in the last round. Based on the updated η values, the decision-making part uses SARSA(λ) algorithm to adjust to the opponent's behaviours in order to get higher payoff.

We present our strategy, discuss the parameter values and analyse their effects on the learning results. The performance of our strategy in the Material Versus Social Payoff Tournament (MVSPT) indicates that our strategy is able to learn and adjust to other strategies' behaviours quickly and accurately and, as a result, earns much higher payoff.

There are several issues worth researching in the future. First, because our η-updating strategy is a fixed-history strategy (see Section 3.1), it can be modelled as an MDP [2]. Therefore, a RL algorithm can be built upon this η-updating process (Section 2.3). We can extend our current single-stage RL-based strategy (where only decision-making part involves RL) to a double-stage RL strategy (where both parts use RL). In addition, since it has been empirically shown in [13] that in the IPD games, by extending the length of the available history in fixed history strategies, the learning performance can be significantly improved. Therefore, we can extend the history length in the η-updating part and check whether the same result applies to DGPD as well.

The second issue is how to initialise Q-values so as to model the environment (the opponent) more accurately. Since Q-value initialisation is equivalent with *potential-based reward shaping* [15], the existing techniques for reward shaping (e.g., [12, 16]) apply to the Q-value initialisation as well. In addition, the techniques that improve the performance of RL by using reward shaping, e.g. Argumentation-Based RL (see [6, 7]), may also apply to this strategy.

The last issue is multi-agent learning in DGPD. Currently we assume that all the other agents have fixed strategies (see Section 3.2), and this is indeed the case in MVSPT. We plan to design more RL-based strategies and observe their performances in the tournament. This will not only improve the multi-agent learning research, but also stimulate research in game theory, economy [10], social psychology [3], etc., since in real economic scenarios, people are always learning from the environment and other people and, as a result, changing their behaviours [5].

7 Acknowledgements

The author wishes to thank Theodore Boyd, Ali ghoroghi and Abbas Edalat for their efforts on designing, testing and organising the Material Versus Social Payoff Tournament. I would also like to thank all the reviewers of the current and previous versions of this paper for providing useful comments and suggestions.

References

1. R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984.
2. M. Babes, E. Cote, and M. Littman. Social reward shaping in the prisoner's dilemma (short paper). In *Proc. of AAMAS*, 2008.
3. C. D. Batson and T. Moran. Empathy-induced altruism in a prisoners dilemma. *European Journal of Social Psychology*, 29:909–924, 1999.
4. P. Chen, B. Keijzer, D. Kempe, and G. Schäfer. The robust price of anarchy of altruistic games. *CoRR*, abs/1112.3680, 2011.
5. A. Edalat, A. Ghoroghi, and G. Sakellariou. Multi-games and a double game extension of the prisoner's dilemma. *CoRR*, abs/1205.4973, 2012.
6. Y. Gao, F. Toni, and R. Craven. Argumentation-based reinforcement learning for robocup keepaway (extended abstract). In *Proc. of COMMA*, 2012.
7. Y. Gao, F. Toni, and R. Craven. Argumentation-based reinforcement learning for robocup soccer keepaway. In *Proc. of ECAI*, 2012.
8. H. Gintis. *The Bounds of Reason: Game theory and the unification of the behavioural sciences*. Princeton University Press, 2009.
9. L. Kaelbling, M. Littman, and A. Moore. Reinforcement learning: A survey. *JAIR*, 4:237–285, 1996.
10. D. Matsumoto, N. Haan, and G. Yabrove et al. Preschoolers moral actions and emotions in prisoners dilemma. *Developmental Psychology*, 22:663–670, 1986.
11. J. Nash. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences*, 36:48–49, 1950.
12. A. Ng, D. Harada, and S. Russell. Policy invariance under reward transformations: theory and application to reward shaping. In *Proc. ICML*, 1999.
13. T. Sandholm and R. Crites. Multiagent reinforcement learning in the iterated prisoner's dilemma. *BioSystems*, 37:147–166, 1996.
14. R. Sutton and A. Barto. *Reinforcement Learning*. MIT Press, 1998.
15. E. Wiewiora. Potential-based shaping and q-value initialization are equivalent. *Journal of Artificial Intelligence Research*, 19:205–208, 2003.
16. E. Wiewiora, G. Cottrell, and C. Elkan. Principled methods for advising reinforcement learning agents. In *Proc. ICML*, 2003.
17. R. Wright. *Nonzero: History, Evolution and Human Cooperation: The Logic of Human Destiny*. Abacus, 2001.
18. R. Wright. *The Evolution of God: The Origins of Our Beliefs*. Abacus, 2010.