A CAS for Finding the Best Strategy for Prisoner's Dilemma

Mirsad Hadzikadic and Min Sun
College of Computing and Informatics
University of North Carolina at Charlotte
mirsad@uncc.edu, msun@uncc.edu

Abstract

Prisoner's Dilemma (PD) is a typical type of non-zero-sum game in game theory. Since first raised by Merrill Flood and Melvin Dresher in 1950's, a lot of research has been done in this area, especially after Robert Axelrod introduced the concept of the iterated prisoner's dilemma in his book "The Evolution of Cooperation" in 1984. In this project, we have explored a complex adaptive system (CAS) designed for finding the "best" strategy for playing PD. The winning strategy depends on the starting conditions. After running the system for a while, an interesting pattern emerges. First, there is an initial consolidation of starting strategies. Second, one (and the same) strategy emerges as the favorite one in the middle of the run. In the end, however, either Tit-for-Tat or "History-Matters" wins the "minds" of the agents. Both of those strategies are of the "cooperate" flavor.

Key words
Game theory, Prisoner's Dilemma, Complex Adaptive System, Self-organization

1. Introduction

Since first raised by Merrill Flood and Melvin Dresher in 1950's [1], a lot of research has been done in the Prisoner's Dilemma (PD) problem, especially after Robert Axelrod introduced the concept of the iterated prisoners dilemma in his book "The Evolution of Cooperation" in 1984 [2]. PD is a typical type of non-zero-sum game in game theory, based on a well-known expression of PD, Canonical PD payoff matrix [2], which shows the non-zero net results for the players.

As opposed to the decision theory, game theory puts more emphasis on the decisions made in the environment in which the players (agents) interact with each other. The choice (action) taken by the opponent will influence the player's choice. In the extended version of PD, the iterated prisoner's dilemma (IPD) [2], two players can choose to either "cooperate" or "defect" during each run. Their decisions depend on the choice made by the opponent during the previous run. The individual and total gains depend on the actions chosen by both players. For example, in the case of players A and B playing "cooperate," the individual gains for A and B is 3 and the total gain is 6, while in the case of person A playing "cooperate" and person B playing "defect," the gain for A is 0, the gain for B is 5, and the total gain is 5.

In a classical PD, individual players try to maximize their payoff by defecting. However, in the IPD with the recorded history players might benefit from cooperating sometime. By cooperating, the player can be sincere (show friendship) or cheat (show fake friendship) the opponent.

Extending the "two-players" to the "many players" problem brings about the situation where hundreds of players play together. Without a central control, players begin to play based on their own initial strategies. After each round, points are added up for each player. Considering different possible reactions and his current strategy, each player decides what action he will make in the next round. In the end, the player with the highest number of points is named the winner.

Given the large number of players (agents) with their interactions, strategies, and fitness function (points), the setting of the Prisoner's Dilemma problem becomes a typical complex adaptive system (CAS).

In this paper, we describe a CAS-based PD system. This approach is different from the traditional ways to establish the "best" strategy for PD: an IPD tournament and evolutionary computation. The IPD tournament [2] only focuses on one-at-a-time comparisons, while the evolutionary computation evolves too many redundant strategies to be useful for heuristic search and ultimate explanation of the results. The CAS approach introduces a quite simple yet explainable way of expressing the strategies for PD. In addition, the influence of all evaluated strategies will be shown as well.

2. Background

2.1. Prisoner's Dilemma

Normally, we use a generalized form to represent the simple Prisoner's Dilemma: Canonical PD payoff matrix [2].

Player A	Player B		
		Cooperate	Defect
	Cooperate	3,3	0,5
	Defect	5,0	1,1

In the classic form of this game, the only possible equilibrium is for all players to defect. But in the Iterated Prisoner's Dilemma the game is played repeatedly, so that each player has the chance to punish or reward others based on strategies that could take into consideration actions taken during the previous play. After a prolonged play, "defect" may not be the dominant choice for the long run.

Finding the strategy to gain the highest number of points is the ultimate problem for the Iterated Prisoner's Dilemma. Every year, the IPD tournament is held to evaluate strategies from different competitors. Some of the known strategies [3] are listed below:

- *Tit-For-Tat* - Repeat opponent's last choice
- *Tit-For-Two-Tats* – Similar to Tit-For-Tat except that opponent must make the same choice twice in a row before it is reciprocated
- *Grudger* - Co-operate until the opponent defects. Then, always defect (unforgiving)
- *Pavlov* - Repeat the last choice if it lead to a good outcome
- *Adaptive* - Starts with the set of pre-selected choices (c, c, c, c, c, c, d, d, d, d, d) but then selects actions which have given it the best average score; re-calculated after each move

In the latest 2004 IPD competition, the winner was the ECS team (a team from the University of Southampton's School of Electronics and Computer Science) [4] that used quite a tricky strategy. The strategy allowed computer agents to collude, rather than to compete with each other. From this experience, we know that the best strategy may not be one that is fixed for all opponents – actions of others and the performance of the whole society of agents may have to be taken into consideration as well. The strategies are affecting each other - each strategy has a different meaning to other players and the "best" strategy might change during a run. Because of this, we decided to put the Prisoner's Dilemma problem in the context of Complex Adaptive Systems.

2.2. Complex Adaptive Systems

A Complex Adaptive System (CAS) is a dynamic network of agents (which may represent cells, species, individuals, firms, nations, etc.) acting in parallel, constantly acting and reacting to what the other agents are doing [5]. The control of a CAS is distributed and decentralized. Any coherent behavior in the system has to arise from competition and cooperation among the agents themselves. The overall behavior of the system is the result of decisions made in each cycle by individual agents [5]. Complex Adaptive Systems enable emergent properties and self-organization. Self-organization is a process in which the internal organization of a system increases in complexity without being guided or managed by an outside source. Self-organizing systems often display emergent properties [6].

This paper describes an approach to implementing the Prisoner's Dilemma problem as a Complex Adaptive System.

3. Experiment Design

In this system, agents represent participants (competitors) in the game. On each turn, agents choose to cooperate or defect. In order to let the agents play repeatedly without the central control, we assign each agent a memory that is used to store information on the results of previous "matches." Agents also "receive" a strategy that is used to decide their actions based on the information they have.

Having been assigned the basic setting and predefined strategy, the agents start moving around and playing with each other. Unlike the usual setting of the Prisoner's Dilemma competition where each competitor (representing one strategy) is asked to compare itself with other agents, one match at a time, in our system a large number of agents are put in one place with randomly assigned strategies. According to our design, for each strategy there are possibly several agents implementing it, with agents playing other agents with different strategies, but also with agents embodying the same strategy. Agents move around in a random fashion and long enough for all of them to have a chance to play a roughly equal number of matches.

NetLogo, a multi-agent programmable modeling tool, was used to build the system [7]. Each agent was described using a chromosome-like structure:

Agent Number	Basic Strategy	Limitation	Reaction1	Reaction2

Where:

- *Agent Number* is the number used to identify each competitor
- *Basic Strategy* is the number indicating the strategy an agent chooses to guide its behavior
- *Limitation* is the number modifying the Basic Strategy. Combined

together, these two numbers define the judgment of situation the agent is facing

- *Reaction1* defines the behavior of the agent if the situation described by Basic Strategy + Limitation applies in the current case/match
- *Reaction2* defines the behavior of the agent if the situation described by Basic Strategy + Limitation does not apply in the current case/match

There are five basic strategies:

0. The agent does not care what happened before
1. The agent takes into consideration the total number of times the opponent cooperated or defected in the past
2. The agent takes into consideration whether during the previous X number of matches/time (X defined by Limitation) the opponent cooperated or defected (X times in a row)
3. The agent takes into consideration the average number of points it got previously by cooperating/defecting when playing against the same opponent
4. The agent takes into consideration whether the number of points it got from the last play is less than three

Reaction1 and Reaction2 can assume one of the following values:

0. Repeat opponent's last action
1. Assume an action opposite to opponent's last action
2. Co-operate
3. Defect
4. Repeat own last action
5. Assume and action opposite to its own last action

For example, competitor 001 shown below simply repeats opponent's last action. This is typical Tit-for-Tat.

001	0	0	0	0

On the other hand, competitor 101 repeats opponent's last action if its opponents cooperate the last two times/matches; otherwise it chooses an action opposite to his own last action.

101	2	2	0	5

As can be seen from the above, our agents do not simply "cooperate" or "defect." They choose to "repeat" or "reverse" an action performed earlier either by their opponents or by themselves. This may be more similar to the way people behave in real life. It also aggregates redundant strategies often present in evolutionary algorithms.

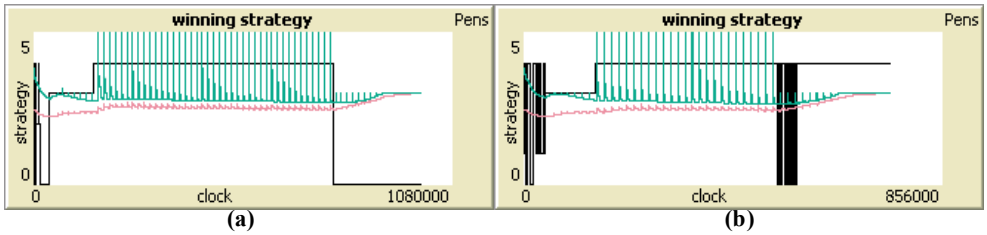
Another parameter, "forgiveness," could be added to the chromosome to represent the random or predefined chance to cooperate (when defecting for a long time) or to defect (to test the opponent after cooperating for a long time). Using "forgiveness" we could represent even a greater variety of strategies.

In NetLogo agents are scattered randomly in the display area (90 * 90 grids) and denoted by a red dot. A strategy is assigned randomly to each agent. Agents also move randomly in the display area. If two agents are in the same neighborhood (8-neighbor grid) a meeting is initiated. Agents play each other based on the strategy they follow and the information they have about each other. After each play, the points are added and agents move on to the next play.

After running the simulation for T times all agents are evaluated using fitness function based on the points collected. Poor performers are replaced by the chromosome of the best performer. This agent evaluation can also be added when two agents meet by having the lesser performer copy the whole or part of the better player's strategy.

4. Experiment Results

We ran the system several times with different initial settings, including the strategies assigned, play order, and distribution of each strategy. The results are summarized in Figure 1. The jagged *green line* shows the highest average-point (winner's point). For each competitor, the *average-point* is calculated as the total number of points gained from the previous plays divided by the total number of plays. The average-point is a number between 0 and 5. The *red line* shows the average of agent averages. It is computed as the sum of average points divided by the number of all competitors. The *average average-point* is a number between 0 and 5 and it outlines the average performance of the whole society. Finally, the *black line* shows the basic strategy chosen by the winner.



(a) **(b)**

Figure 1: "Cooperate" as the winning strategy

In the two graphs of Figure 1 (a and b), representing two different runs of the system,

although the final winning strategies are different, the green line and the red line converge into one line at the exact number three, which shows that in the end the highest average-point equals the average average-point. In other words, in the end, all competitors cooperate and gain three points after each run.

Even the winning strategies are essentially similar to each other. The final winning strategy for the case represented in Figure 1(a) is the agent with the chromosome

	0	0	0	0

which is a typical Tit-for-Tat strategy: repeat opponent's last action. A competitor with this strategy reacts quickly to the action of others. It is quite defensive when it meets a defector, but friendly when it meets a collaborator. In the beginning of the run, Tit-for-Tat can be neither the best nor the worst strategy. However, when the whole society becomes friendlier, this strategy has a chance to be both the best and the dominating one.

In the graph represented in Figure 1(b), the final winning strategy is the agent with the chromosome

	4	-1	5	0

The agent looks at the points it earned from the last play and if it received three or more points (it gets three points if they both cooperate, or five points if it defects while the opponent cooperates), it chooses to repeat the opponent's last action (cooperate). However, if it earned less than three points from the last play (zero if it cooperates while the opponent defects, or one if they both defect), the agent chooses the action opposite to its own last action. Similar to the Tit-for-Tat player, the player with this "History Matters" strategy is friendly as soon as the opponent cooperates, and it punishes opponents quickly when it feels betrayed. The only difference is that "History Matters" shows the willingness to cooperate first when it notices that both players are in the situation of defection and it will try to end the "lose-lose" situation. This approach helps it become a winner earlier than the Tit-for-Tat does.

Our experiment also indicates that players with different strategies not only compete with each other but they are also necessary components for building the whole society. For example, the final winning strategy is almost never the initial point leader. It is usually in the middle of the pack at first, but it demonstrates its power when the whole society develops preference for cooperation. Strategy number 3 is important in terms of its ability to change the society from a hostile society into a friendly one. From Figure 2 we can see that the average average-point began to go up as the highest average-point inched down, marking the time when Strategy 3 became the dominant strategy in the society. A player with Strategy 3 cares only

about the average points it earned previously be either cooperating or defecting – a relatively stable player who does not change quickly. This makes it a "society changer." Without the participation of players with Strategy 3, Tit-for-Tat or "History Matters" would hardly end up being the best strategies.

5. Conclusions and Future Work

In this paper, we describe the use of a complex adaptive system to find the optimal approach to solving the Iterative Prisoner's Dilemma. The simple representation mechanism we deployed used only five "genes" to implement all previously reported strategies, in addition to introducing many new ones. The agents adopted strategies randomly and then played each other until a winning strategy emerged as the "consensus" strategy in the society of agents. Some agents were given the ability to record outcomes of a randomly selected number of previous matches. Consequently, the society included agents with both differing and similar strategies (e.g., identical strategies with parameters like "depth of memory" slightly modified) that allowed various strategies to be tested against each other, including themselves.

The results have shown that:

- The "first-principles-based" knowledge representation (general and flexible) allows us to produce and test "all possible" strategies within the given representational framework
- A "consensus" (cooperation) strategy emerges after a long run, even though it is not always exactly the same one
- All winning strategies have a similar pattern during the run
- All strategies contribute to the "winning" strategy (context dependency)

In the future, we will extend this research to aggregated strategies in hope of demonstrating the "wisdom of the crowd" phenomenon.

6. References

[1] Flood, M.M. (1958) "Some Experimental Games, Research Memorandum RM-789, RAND Corporation," Management Science (5)
[2] Robert Axelrod, 1984, "The Evolution of Cooperation" New York: Basic Books
[3] http://www.iterated-prisoners-dilemma.net/prisoners-dilemma-strategies.shtml
[4] http://www.prisoners-dilemma.com/results/cec04/ipd_cec04_full_run.html
[5] M. Mitchell Waldrop, 1992, "Complexity: The Emerging Science at the Edge of Order and Chaos"
[6] http://www.wikipedia.org/
[7] http://ccl.northwestern.edu/netlogo/