Assignment Week 8

Philosophy 444

Due 12.01pm, 25 February, 2023

Iterated Prisoners' Dilemma

We are going to run a tournament pitting different strategies against each other in Iterated Prisoners' Dilemma. Your task is to write a strategy, and write a short document (about 300 words) explaining why you picked that strategy. Both of these are due **Friday, March 12** at **5pm**. You can use the Canvas site to collaborate with others on ideas, but you must submit your own work. You will be graded in part on how well you do in the tournament, and in part on how well you defend your strategy choice.

In the tournament, your strategy will be matched up in turn with each of the strategies submitted (including your own), plus five pre-defined strategies. You play 'against' each of these others five times. Each play will consist of roughly 200 rounds of Prisoners' Dilemma (PD). Each round of PD consists of a game with the following payouts, in points. (You are Row, the other player is Column.)

	C	D
C	3, 3	0, 5
D	5, 0	1, 1

o you will play each 'opponent' roughly 1000 times. The number of points you each get over those 1000 games will be added to your individual scores, and then you'll go on to play another player. At the end of the tournament, when everyone has played everyone, whoever has accumulated the most points wins.

Note that there is a bit of zero-sum-ness to the game at the end. While you are playing any individual player, it is not a zero-sum situation; there are ways for both of you to improve or detract from your overall position. But your ultimate aim is not just to get a lot of points, it is to get more points than anyone else in the class.

Writing a Strategy

To explain to the computer how you want to play, you need to write a short bit of code. The language we'll use for this is fairly basic - essentially you will describe a **Finite State Machine**.

Your machine will at any time be in one of a number of *states*. There will be finitely many of these - hence the name. A state has three key characteristics.

1. Whether the machine plays C or D in that state.
2. What state the machine moves into if the opponent plays C.
3. What state the machine moves into if the opponent plays D.

One of these states, state 0, will be the initial state of the machine. So here is a very simple machine.

Number	Play	Move if C	Move if D	Notes
0	C	0	1	Cooperate state
1	D	0	1	Defect state

This machine will play C at first, since that's the play in state 0. After that, it mimics what the previous player has done. No matter what state it is in, it will move into cooperate state if the other player co-operates, and move into defect state if the other player defects. This strategy is known as **Tit-for-Tat**, and is one of the two pre-defined strategies I mentioned above.

Let's look at a slightly more complicated strategy.

Number	Play	Move if C	Move if D	Notes
0	C	0	1	Cooperate state
1	C	0	1	After single defection
2	D	0	2	After multiple defections

This strategy will start cooperating, then will defect any time the opponent has defected on the previous two turns. Here's how it does that. State 0 is cooperate, so it starts off cooperating. It is the base state; think of it as being happy. State 1 is when the opponent has defected once. It's what happens when (and only when) the machine is happy, then sees its opponent defect. In state 1, the machine still cooperates, but it is vigilant. From now on, any more defections will send it to state 2, where it defects. But at any stage, a single cooperation from the opponent will send it back to happy.

Finally, a strategy that keeps count of things. The idea is to play Tit-for-Tat, but with a fuse. Once the opponent defects four times, we defect forever. So we need to keep count of defections. We do that by having more states. In the notes, I'll list the count of defections.

Number	Play	Move if C	Move if D	Notes
0	C	0	1	Base; count=0
1	D	2	3	Defect; count=1
2	C	2	3	Coop; count=1
3	D	4	5	Defect; count=2
4	C	4	5	Coop; count=2
5	D	6	7	Defect; count=3
6	C	6	7	Coop; count=3
7	D	7	7	Defect; count=4

The machine really cares about two things; how many times the opponent has defected, and what the opponent did on the previous play. At each time (until it loses patience at the end), it responds to cooperating with cooperating and staying in place, and it responds to defecting by defecting and moving closer to the lose patience state. Once it gets to state 7, it stays there; no matter what the opponent does it will still defect.

For this assignment, you should submit two things. Your program should be a plain text file (saved as .txt on your computer) with the following structure.

- The first three lines are your name, your strategy's name, and the number of states in your machine.
- Each line after that is a state. It should say what move to make in that state, then a comma, then what state to move to if the opponent cooperates, then another comma, then what state to move to if the opponent defects.

Here, for instance, is how I would submit a file if I was playing Tit-for-Tat.

Brian Weatherson
Very Original Strategy
2
C, 0, 1
D, 0, 1

You should also write a note - roughly 300 words long - about what your strategy aims to do, and why you chose it. And make sure that your program has the right syntax; we can't debug programs. If you want to check for sure, install the Oyun software https://oyun.charlespence.net and run the program yourself to see.

We'll compile all the strategies over the weekend, and next week we'll discuss how they did. Note that the tournament will include five pre-defined strategies:

1. Tit-for-Tat
2. Random, who plays randomly.
3. All C, who always cooperates.
4. All D, who always defects.
5. No forgive, who cooperates until you defect, then always defects.