CE/CZ4046 INTELLGENT AGENTS

Assignment 2

知己知彼, 百战百胜

Lim Song Wei, Greg U2120517G

Notation

C is Cooperate; D is Defect

CCD – Player 1 Cooperates, player 2 Cooperate, player 3 Defect

Player 1 is always the player the point we are tracking unless otherwise stated

Agent Design

Key Observations and Strategies

First step in creating my agent was to analyses agent that perform well and understand their strategy. From this, here are some of the key strategies and observations I made:

0 – Zero-sum Game

Though the Prisoners Delima itself is not a zero-sum game, the eventual score of the game is your relative position to other players, and relative position is a Zero-sum game, therefore minimizing points given to other strategies is important factor to note.

1 - Friendly strategy

Strategies that do not defect first perform much better than those that defect first. This is only true in general defects and not for end game defects.

2 - Friendly defect

It is observed that most strategies, CDD results in player 1 defecting next round.

It is observed that CCD results in 3 points, but DDD results in 2 points.

Therefore, it is better to loop in a CCD with than to transition into DDD. 3>2.

Therefore, the main strategy I use only defects if both players defect.

3 - Last round defecting

A common strategy is to defect on the last round when the opponent cannot defect. For this set of rules, the last round could be $n \mid n \in [90,\ 109)$.

For my strategy, I implement a fixed defect at round 108 and 109. In most games, the game would end before round 108. This value was obtained through testing.

4 – Defecting time

From testing, the more players there are, the later you should start defecting. This was suggested when testing.

My theory: When there are less players, pulling the opponent score down is beneficial, but when the number of players increases, the effectiveness decreases and the self-harm increases.

5 – Fixed Defecting Time

A relatively common strategy I noticed was random last rounds defecting, where the agent will pick a random time to switch to defect in the last 20 rounds.

Theorem: Having a random round to defect in the last few rounds is worse than a fixed round to defect.

Assumption: Defecting in the endgame would result in retaliation

Observation:

- Players always play one game with 3 of their same strategy, and n-1 games with 2 of their same strategy.

Proof: For this proof, I would show that fixed time to defect is equal or better than random time to defect for all round types.

Case 1: all 3 players are the same strategy

CCC->6+6+6=18, CCD->3+3+8=14, CDD->0+5+5=10

18>14 and 18>10

Therefore, defecting at a fix time instead of random time is better for your strategy

Case 2: 2 players are the same (first 2 is your strategy)

CCC->6+6=12, CCD->3+3=6 (both of your strategy defect at same times)

CDC->3+8=11, CDD->0+5=5 (both of your strategy defect at different times)

12>11 and 6>5

Regardless of whether the third player defects, the 2 same strategies should CC

Therefore, the same strategy that always defects at the same time will perform better.

Case 3: 1 player against 2 other strategies

There may or may not be an optimal value for when to start defecting. Since each game of 3 player is episodic, random and fixed have no difference form an episode frame of view. Therefore, it is not possible for random to outperform fixed defecting strategy in case 3.

Since all cases are covered and show that fixed strategy is better or equal random, my theorem is proven.

Agent Strategy

1. IF round 0: C (friendly strategy)
2. IF round is 109: D (last round defecting by default)
3. IF round > 107: D (109 is the last round, fix defect strategy, late defect strategy)
4. IF player 1 defect more than 40% and player 2 defect more than 40%: D (protect against random and always defect)
5. IF both defect: D (friendly defect strategy, do not defect if only 1 defect)
6. ELSE: C (friendly strategy)

Agent Implementation

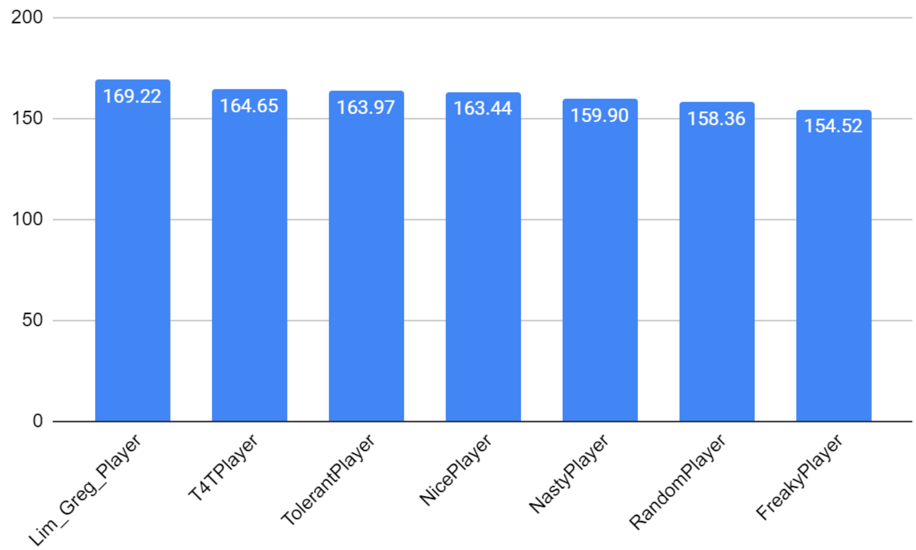
Initializing default values. These values can be optimized.

```
// when to start defacting
// testing suggesting defecting only the last 2 rounds to be the most effective
// defecting early make the score less consistent and is harmful
final int DTime = 108-1; // 109 -> defact on last round
```

The code is a one for one implementation of the strategy above.

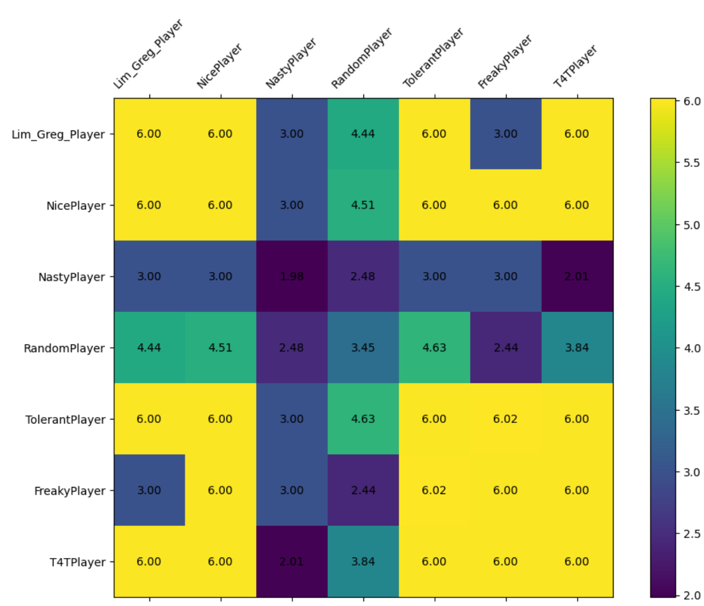
```
// Friendly
if (n==0) return 0;

// Last round defect: defect with no punishment
if (n >= 109) return 1;

// Defecting Time: defect with minimal punishment
if (n > DTime) return 1;

// To detect if both players are defecting often. could suggest a random
if(n>5){
    int oppNumDefections1=0;
    int oppNumDefections2=0;
    for (int index = 0; index < n; ++index) {
        oppNumDefections1 += oppHistory1[index];
        oppNumDefections2 += oppHistory2[index];
    }
    if(oppNumDefections1/n > 0.4 && oppNumDefections2/n > 0.4) return 1;
}

// Assume if both defects, it cannot be recovered
// Friendly defect, only D when both D
if (oppHistory1[n-1] == 1 && oppHistory2[n-1] == 1) return 1;

// Friendly default
return 0;
```

Performance Evaluation



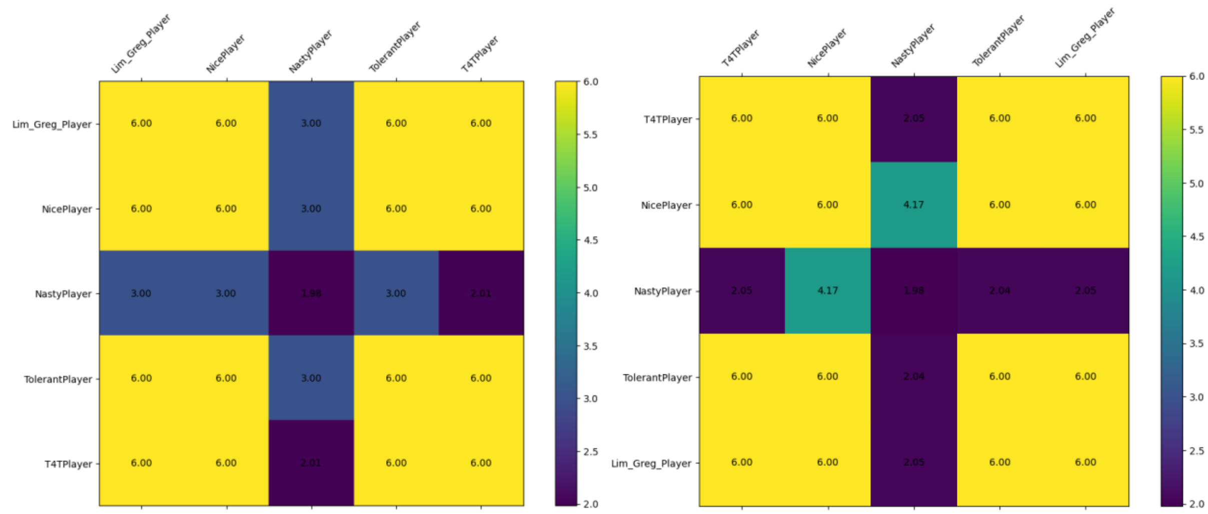
From the bar chart, it shows the revertive difference between my strategy and the 6 given strategies. From this, it can be observed that friendly strategies then to perform better. With random strategies and unfriendly strategies performing worse.

Though my strategy got the highest score this is not representative as most real strategies are friendly and minimally random.



The matrix map above shows how Player 1, my player, performs against other pairs of players. The x-axis is player 2 and the Y-axis is player 3.

Performance Analysis



LHS: how my code performs. RHS: how T4TPlayers perform

When comparing how my strategy compares to T4TPlayer (the second best), our strategy diverges in how we handle nasty players. On the left, when my strategy meets P2 (my strategy) and P3 (nasty), both p1 and p2 cooperate getting 3 points each and nasty get 8 points. But for T4TPlayer, all 3 players enter a defect loop gaining 2 points a round. T4TPlayers exploits always nice strategies.

Analyzing interactions

Against 2 others friendly strategies, my strategy will cooperate. CCC -> CCC

Against 1 always nasty and 1 always friendly strategy, my strategy will assume that the friendly strategy will defect if the other 2 defect (assume DDC -> DDD), hence I would not defect to ensure the friendly strategy does not defect. CDC -> CDC, Resulting in a payout of P1(me): 3, P2(nasty): 8, P3(nice): 3

Against 2 others nasty strategy, my strategy is quick to defect to reduce points lost. CDD -> DDD.

Conclusion

To conclude, an effective strategy should be nice, retaliate quickly, forgive and aim to maximize points instead of having more points. Additionally, from my observations and analysis, last round defecting is a default and last few rounds defecting is viable; It is better to have CCD than DDD thus only defect if both defects, and having a fix defecting strategy is better. From these observations and analysis, as well as testing against other strategies I found or created, this strategy should be well optimized.