Prisoner's Dilemma with Optional Cooperation and N Participants

Matt Lee
TJHSST Computer Systems

Abstract

This project is designed to simulate the classical Prisoner’s Dilemma with a large number of participants and set options to cooperate with others or not. The purpose of this project is to allow the Prisoner’s Dilemma to have variable parameters so that a variety of situations and settings could be tested. The result that is expected is a variety of simulations that will show how a specific situation can turn out when given options to cooperate, backstab, or ‘join forces’.

Background

The Prisoner’s Dilemma has been implemented a large number of times. There have even been competitions held to see who could make an algorithm that would maximize the payout for their specific participant. As stated by Robert Axelrod, the author of ‘The Complexity of Cooperation’ the best strategy for maximizing payoff is to use ‘tit for tat’. Tit for tat is a strategy where the participant mimics the last move played by the opponent, which in the long run, enables the user to maximize his payout at the end of the ‘game’ of Prisoner’s Dilemma. However, interestingly enough, when both participants initiate tit for tat, it doesn’t become the optimal strategy. From here, a large number of variations have been made to the Prisoner's Dilemma, including implementing “N” participants instead of just two.

```
-   if(turn!=0)
            {

               while(run<size)
               {
                  if(run!=IDtag)
                  {
                     Boolean
    desu=(Boolean)list.get(run);
                     boolean desu2=desu.booleanValue();
                     if(desu2==false)
                        falsers++;
                     else if(desu2==true)
                        truers++;
                  }

                  run++;
               }
               if(falsers>=truers)
                  player.setDecision(false);
               else if(falsers<truers)
                  player.setDecision(true);
            }
            else
               player.setDecision(true);

            /*while(counter<size)
            {
               prisoner player2=(prisoner)
    list.get(counter);
               if(player2==player)
               {
                  break;
               }
               else
               {
                  boolean oppDeci=player2.getDecision();
                 player.setDecision(oppDeci);
               }
```

Progress

The options to choose strategies and make a number of opponents have been added to the program. The Prisoner class has undergone some changes including changing string identification to integer identification and the addition of a new method. The program itself has several variations of Tit for Tat added into it as well at this point.

Results

The results I expect are a variety of scenarios to be implemented by the Prisoner's Dilemma that will include “N” participants of the user's choice and option to enable morality.