Genetic Approach to Find Optimal Moves for Prisoner Dilemma Game

Navdeep Kaur Dhillon
M. Tech, Department of Computer Science and Engg.
Guru Nanak Dev University
Amritsar,Punjab, India
navi.dhillon03@gmail.com

Navdeep Kaur Maan
M. Tech, Department of Computer Science and Engg.
Guru Nanak Dev University
Amritsar,Punjab, India
navi26maan@gmail.com

Abstract—The Prisoner‘s Dilemma, a simple two-person game invented by Merrill Flood & Melvin Dresher in the 1950s, has been studied extensively in Game Theory. Till now the work has been done to study the environment in which strategies for the Prisoner's Dilemma can be evolved through Genetic Algorithm approach and find out best strategy for playing game. We have applied genetic approach to find out best moves taken by two players in prisoner dilemma game. There are two traits in this problem: the ability to defend against defectors, and the ability to cooperate with other cooperators . First the algorithm to play iterated prisoner dilemma game for two players is proposed. Then various moves of two players in three games are studied and through genetic approach the best moves are find out. In the end results of implementation of prisoner dilemma with genetic approach are given.

Keywords: prisoner dilemma game; Algorithm for two player game; genetic approach.

I. INTRODUCTION

The Prisoner's Dilemma can be formulated as follows [4]: Two individuals (call them Mr. X and Mr. Y) are arrested for committing a crime together and are held in separate cells, with no communication possible between them. Mr. X is offered the following deal: If he confesses and agrees to testify against Mr. Y, he will receive a suspended sentence with probation, and Mr. Y will be put away for 5 years. However, if at the same time Mr. Y confesses and agrees to testify against Mr. X, his testimony will be discredited, and each will receive 4 years for pleading guilty. Mr. X is told that Mr. Y is being offered precisely the same deal. Both Mr. X and Mr. Y know that if neither testifies against the other they can be convicted only on a lesser charge for which they will each get 2 years in jail. Should Mr. X "defect" against Mr. Y and hope for the suspended sentence, risking, a 4-year sentence if Mr. Y defects? Or should he "cooperate" with Mr. Y, in the hope that he will also cooperate so each will get only 2 years, thereby risking a defection by Mr. Y that will send him away for 5 years?

In this paper we make the following contributions:

1. We prepared algorithm for iterated prisoner dilemma game for two players.
2. We proposed genetic algorithm to find best moves that can be taken by players while playing iterated prisoner dilemma game.
3. We study the results of algorithm by implementing algorithm in java and verify outputs .

Our main results are as follow

1. The output shows the best moves that two players of prisoner dilemma game can take to increase their scores in game.

II. PRISONER DILLEMA GAME

The game can be described more abstractly[2]. Each player independently decides which move to make— i.e., whether to cooperate or defect. A "game" consists of each player's making a decision (a "move"). The possible results of a single game are summarized in a payoff matrix like the one shown in table Here the goal is to get as many points (as opposed to as few years in prison) as possible. (In table , the payoff in each case can be interpreted as 5 minus the number of years in prison.) If both players cooperate, each gets 3 points. If player A defects and player B cooperates, then player A gets 5 points and player B gets 0 points, and vice versa if the situation is reversed. If both players defect, each gets 1 point. What is the best strategy to use in order to maximize one's own payoff? If you suspect that your opponent is going to cooperate, then you should surely defect. If you suspect that your opponent is going to defect, then you should defect too. No matter what the other player does, it is always better to defect. The dilemma is that if both players defect each gets a worse score than

if they cooperate. If the game is iterated (that is, if the two players play several games in a row), both players always defecting will lead to a much lower total payoff than the players would get if they cooperated.

TABLE 1: PAYOFF MATRIX FOR PLAYERS

	B cooperates	B defects
A cooperates	R=3, R=3	S=0, T=5
A defects	T=5, S=0	P=1, P=1

Let *CC* be the number of points won by each player when they both cooperate; let *DD* be the number of points won when both defect; let *CD* be the number of points won by the cooperating party when the other defects; and let *DC* be the number of points won by the defecting party when the other cooperates.

Then the prisoner's dilemma situation is characterized by the following conditions:

- DC > CC > DD > CD
- CC > (DC+CD)/2.

The gain from mutual cooperation is greater than the average score for defecting and cooperating.

Three strategies are used to play iterated prisoner dilemma[1]:

- **Tit for Tat(TFT) Strategy:** The player will cooperate at first iteration. In next iteration opponent's move in previous iteration is assigned to player.
  TFT strategy played by player is as follows:
  If CC(Case1), then C,
  If CD(Case2), then D,
  If DC(Case3), then C,
  If DD(Case4), then D.
- **Paylov Strategy:** Player will cooperates at the first iteration and whenever the player and opponent did the same thing at the previous iteration; Pavlov defects when the player and co-player did different things at the previous iteration
  Pavlov strategy played by player is as follows:
  If CC(Case1), then C,
  If CD(Case2), then D,
  If DC(Case3), then D ,
  If DD(Case4), then C.
- **Random Strategy:** player can either cooperate of defeats its opponent by taking any random move.

III. RELATED WORK

The political scientist Robert Axelrod pioneered research in this field in the late 1970's [4]. As part of research Axelrod held a series of computer tournaments in which IPD strategy entries from around the world played games of the IPD against one another in a round robin fashion. This tournament aimed to identify optimal strategies for the Prisoner's Dilemma (there is no best strategy; the success of a strategy depends on the other strategies present). The Tit For Tat strategy (TFT) won both computer tournaments conducted by Axelrod indicating that it is an optimal strategy. This strategy simply cooperates on the first move and then only defects if the other player defected on the previous move. It is interesting to note that the TFT strategy can never obtain a higher score than its opponent. However this is unimportant as the Prisoner's Dilemma is a nonzero sum game, you do not have to do better than your opponent; you have to use your opponent to get a high score yourself.

Axelrod found that a strategy normally has four different characteristics to be successful in a Prisoner's Dilemma tournament:

- Nice – Cooperates on first move
- Retaliatory – will defect if defected against
- Forgiving – can be made to cooperate after starting to defect
- Clarity – don't be too complex

TFT possesses all of these characteristics.

IV. ALGORITHM FOR ITERATED PRISONER DILEMMA

For designing algorithm for prisoner dilemma game two players p1 and p2 are considered and game is played between them n time. Player p1 uses tit for tat strategy and player p2 uses random strategy. Player p1 will cooperate at first move and player p2 can either cooperate or defeat. If both players cooperate with each other they will get reward of 3 score each. If player p1 defeat and player p2 cooperate, p1 will get temptation and 5 score and p2 will get sucker's payoff of 0 score. If both players defeat each other , they will get penalty of 1

score. At the end of first iteration player p2 move is assigned to player p1 according to rule of tit for tat strategy. the end of game total scores of both players is calculated and player with high scores is declared as winner.

The algorithm for iterated prisoner dilemma for two players is as follow:

```
Algorithm Prisoner_Dilemma (p1, p2)
  //p1 and p2 are two players of game
   //0:=defect and 1:=cooperate are two moves of players
   //5:=temptation, 3:=reward, 1:=penalty, 0:=sucker's payoff
 {
        p1move:=1;                              // p1 will cooperate at start
   For i: =1 to n do
    {
      p2move:=Random (0, 1);              //p2 can take any random move
      if((p1move=1)and(p2move=1)) then     //both players cooperate  with each other
      {
        p1score:=3;
        p2score:=3;                                //both players get reward
      }
     Else if((p1move=0)and(p2move=1)) then      //  p1 defect  p2
       {
        p1score:=5;
        p2score:=0;
}
     Else if((p1move=1)and(p2move=0) )then
       {
        p1score:=0;
       p2score:=5;
        }
     Else if((p1move=0)and(p2move=0)) then
       {
         p1score:=1;
        p2score:=1;
        }
     p1totalscore+=p1score;
     p2totalscore+=p2score;
     p1move:=p2move;                    //p2move will be assigned to p1 move
  }
    print p1totalscore;
    print p2totalscore;
     if(p1totalscore>p2totalscore)      //find winner at the end of game
     {
     print("p1 is winner");
      }
     else
     {
   Print("p2 is winner");
    }
}
```

V. GENETIC ALGORITHMS

Genetic Algorithms are search algorithms based on the mechanics of natural selection and natural genetics[5]. They usually work by beginning with an initial population of random solutions to a given problem. The success of these solutions is then evaluated according to a specially designed fitness function. A form of natural selection is then performed whereby solutions with higher fitness scores have a greater probability of being selected. The selected solutions are then mated using genetic operators such as crossover and mutation. The children produced from this mating go on to form the next generation. The theory is that as fitter genetic material is propagated from generation to generation the solutions will converge towards an optimal solution. This research utilizes Genetic Algorithms to search optimal moves to be taken by two players for iterative Prisoner's Dilemma.

A simple GA works on the basis of the following steps:

Step 1

Start with a randomly generated population of n *l*-bit chromosomes (Candidate solution to a problem).

Step 2

Calculate the fitness *f(x)* of each chromosome *x* in the population.

Step 3

Repeat the following steps until n offspring have been created:

- Select a pair of parent chromosomes from the current population, the probability of selection being an increasing function of fitness. Selection is done ―with replacement meaning that, the same chromosome can be selected more than once to become a parent.
- With probability *Pc* (the ―crossover probability), crossover the pair at a randomly chosen point to form two offspring. If no crossover takes place, form two offspring that are exact copies of their respective parents.
- Mutate the two offspring at each locus with probability *Pm* (the mutation probability or mutation rate), and place the resulting chromosomes in the new population.

Step 4

Replace the current population with the new population.

Step 5

- Go to step 2.

VI. GENETIC ALGORITHM IN PRISONER DILLEMA

In this paper the game between player1 and player2 is played for three times. So the total number of strings produced is six. Then the scores of both players are calculated by playing the three chromosomes of player1 with three chromosomes of player2. The scores gained by both players are considered as fitness. The sum fitness is sum of all scores of player1 and player2.then the expected count is calculated by dividing the individual fitness by average. The selection of best six chromosomes is taken according to top three expected counts. These six chromosomes are put into mating pool. Then according to crossover probability the crossover point is selected between total length of chromosome. After crossover the mutation of chromosomes take place according to mutation probability and mutation point. The next generation is produced after mutation . Again the fitness of both players are calculated and sum fitness and average fitness is checked to check improvement in next generation. The flow chart given below show the working of genetic algorithm in prisoner dilemma game:

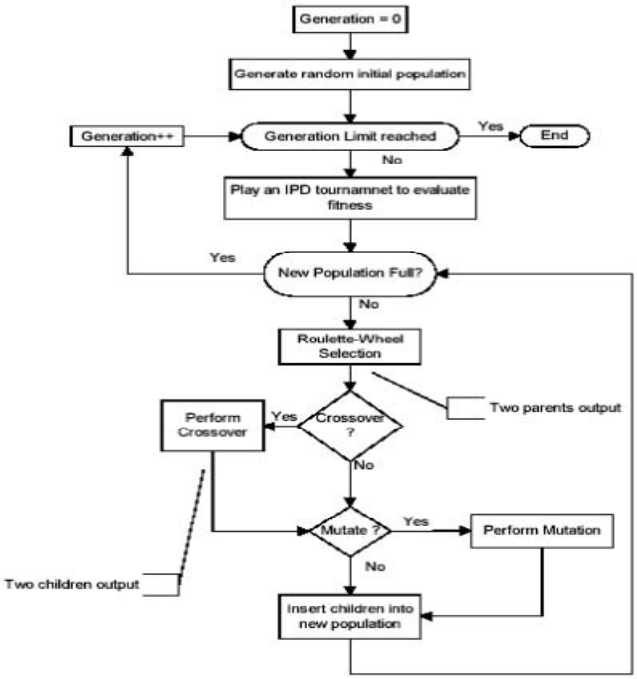

Figure1: Working of prisoner dilemma through genetic approach

VII. OUTPUTS OF IMPLEMENTATION

1)Creating initial population: The initial population is created by taking six strings of random integers between 0 and 1 where first three string presents the moves taken by payer 1 in three games and next three strings presents the moves taken by player2 in three games.

```
***GENERATING INITIAL POPULATION***
        Initial population of player 1 in 3 games

11110
01010
11011
        Initial population of player 2 in 3 games

10101
00100
11011
```

Figure 2: Initial population

2)Calculating fitness: The fitness is calculated on the basis of scores. If moves of player1 and player2is 0,both will be given 3 score. If moves of player1 and player2 is 0 and 1 respectably, then player1 will be given 0 score and player2 will be given 5 scores. If Moves of player1 and player2 is 5 and 0 respectably, then player1 will be given 5 score and player2 will be given 0 scores. If moves of both players are 1 then they will be given 1 score. The scores are calculated by playing game between each chromosome of player1 with each chromosome of player2. Then the total score obtained by chromosome pair are taken as fitness. The total fitness is calculated by adding fitness of all pairs and average fitness is calculated by dividing total fitness by total no. of chromosome.

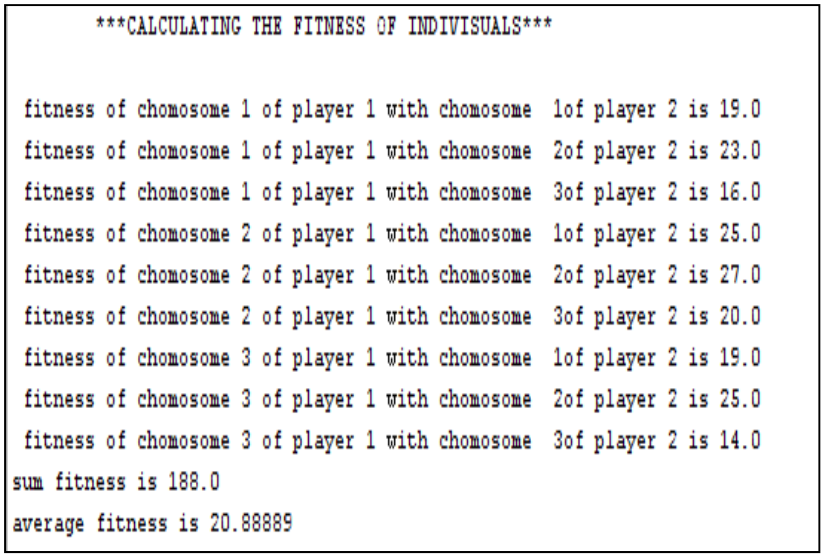

Figure 3: Fitness of Individuals

3)Selection: The selection takes place according to expected count. The expected count is calculated by dividing the fitness of each chromosome pair with average fitness. Then the three chromosome pairs with high expected counts are selected and put into mating pool.

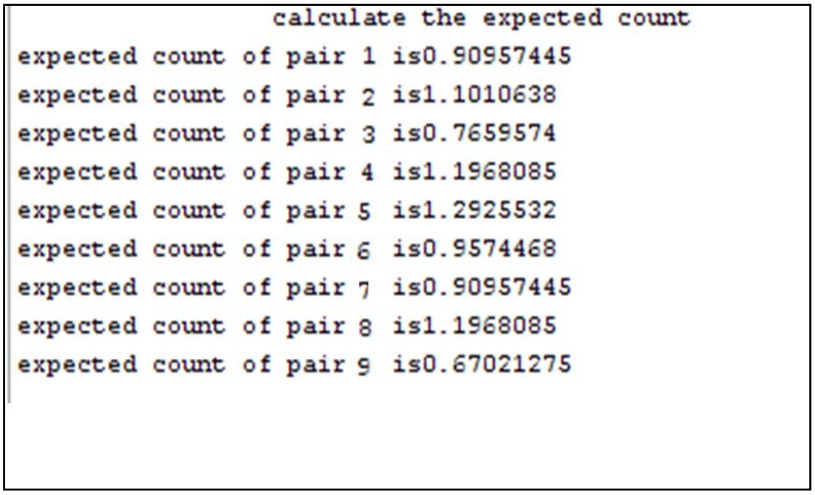

Figure 4: Expected Count of Chromosome pairs

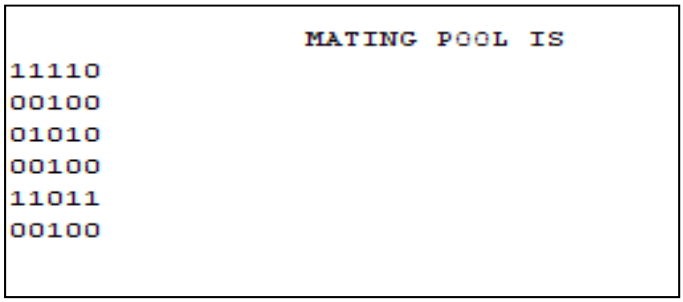

Figure 5: Mating pool

*4)Crossover :*If the crossover probability is not zero, the crossover point is find out between total length is find out. Then the crossover between parent chromosomes takes place as shown in figure. In figure 6,first the children produced for player1 are printed and then children produced for player2 are printed.

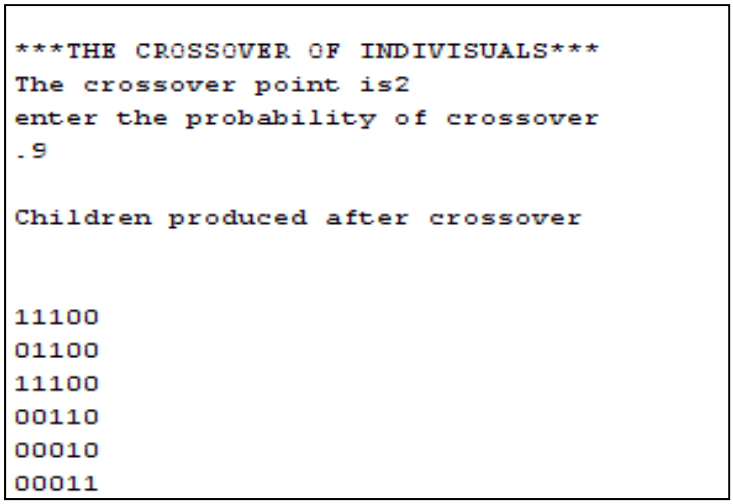

Figure 6: crossover between Parent Chromosomes and new Childs produced

Mutation: Now the mutation probability is checked. If it is non zero the mutation point is find out between total length of chromosome and bit at that mutation point will be flipped. Figure 7 shows the process of mutation.

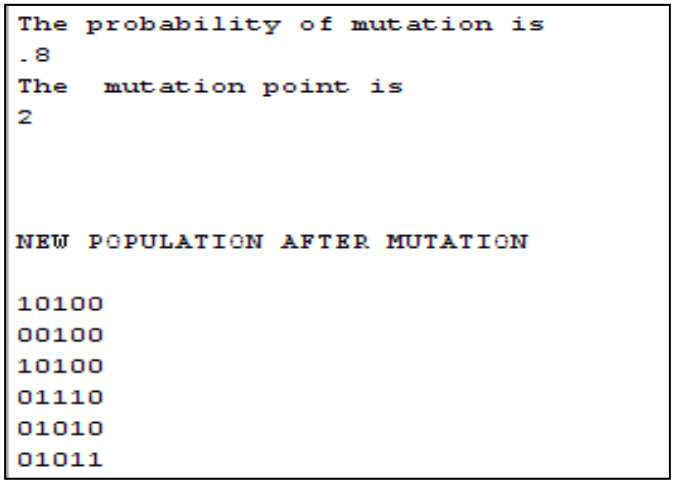

Figure 7: Mutation of chromosomes

5)Replacement: The fitness of next generation is again find out to see the improvement in scores, sum fitness and average fitness. If there is improvement in sum fitness and average fitness, the generation produced after crossover and mutation replace the previous generation. In figure 8, first three string presents chromosomes of player 1, and next three strings presents chromosomes of player 2.

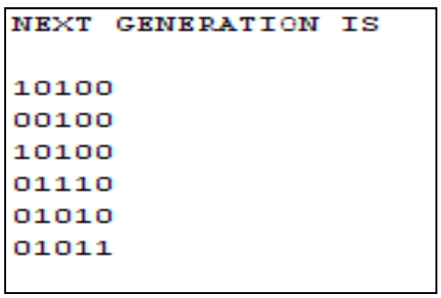

Figure 8:Next generation produced

```
 fitness of chomosome 1 of player 1 with chomosome  1of player 2 is 23.0
 fitness of chomosome 1 of player 1 with chomosome  2of player 2 is 26.0
 fitness of chomosome 1 of player 1 with chomosome  3of player 2 is 25.0
 fitness of chomosome 2 of player 1 with chomosome  1of player 2 is 24.0
 fitness of chomosome 2 of player 1 with chomosome  2of player 2 is 27.0
 fitness of chomosome 2 of player 1 with chomosome  3of player 2 is 26.0
 fitness of chomosome 3 of player 1 with chomosome  1of player 2 is 23.0
 fitness of chomosome 3 of player 1 with chomosome  2of player 2 is 26.0
 fitness of chomosome 3 of player 1 with chomosome  3of player 2 is 25.0
sum fitness is225.0
average fitness is 25.0
```

Figure 10: Fitness, sum fitness, average fitness of next generation

VIII . CONCLUSION

This paper presents the algorithm to play iterated prisoner dilemma game for two players .and genetic approach to find the optimum moves for game. The chromosomes are taken from the six moves of two players in three games. The chromosomes with less expected count are discarded. The next generation produced after crossover and mutation is checked for improvement by calculating the sum fitness and average. If there is any improvement in fitness, of individuals, sum fitness and average in next generation then next generation replace previous generation.

REFERENCES

[1] Jennifer Golbeck, " Evolving Strategies for the Prisoner's Dilemma", Computer Science Department University of Maryland, College Park, MD USA; golbeck@cs.umd.edu http://www.cs.umd.edu/~golbeck.

[2] Adnan Haider, "Using Genetic Algorithms To Develop Strategies For Prisoner Dilemma", *PhD* Fellow, Department of Economics Pakistan Institute of Development Economics Islamabad, Pakistan.

[3] Andrew Errity , "Evolving Strategies for the Prisoner's Dilemma Technical Manual", Faculty of Computing and Mathematical Sciences, Dublin City University.

[4] Robert Axelrod, "The Evolution of Cooperation", Penguin Books, 1990.

[5] David E. Goldberg, "Genetic Algorithms in search, optimization, and machine learning", Addison-Wesley Publishing, 1989