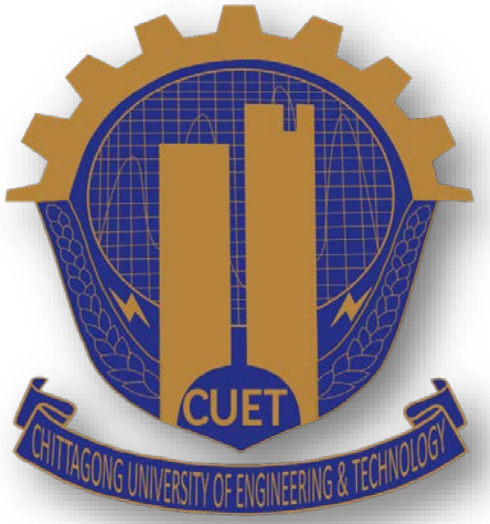
DEPERTMENT OF COMPUTER SCIENCE & ENGINEERING



Name of the project: Connect4 game with AI

Course no: CSE-300

Course title: Software Development Project (sessional)

Date of submission: 09.02.2020

Submitted by:

ID: 1604070, 1604089

Level: 3

Term: 2

Section: B

Group: B1

Submitted to:

Dr. Md Iqbal Hasan Sarker (Assistant Professor)

Animesh Chandra Roy (Assistant Professor)

Md. Billal Hossain (Lecturer)

INDEX

Chapter A

1. Introduction:

Artificial intelligence (AI)**,** the ability of a digital computer or computer-controlled robot to perform tasks commonly associated with intelligent beings. The term is frequently applied to the project of developing systems endowed with the intellectual processes characteristic of humans, such as the ability to reason, discover meaning, generalize, or learn from past experience.In computer science, artificial intelligence (AI) is intelligence demonstrated by machines, in contrast to the natural intelligence displayed by humans. It is often used to describe machines (or computers) that mimic "cognitive" functions that humans associate with the human mind, such as "learning" and "problem solving". In our project, we had made Connect4 game with AI. Board game is a vast area in AI. So on that consideration, we had tried to develop a board game with AI.

2. Background and present state:

A tradition was to play board game by self-playing. But it is not efficient enough for building a board game. Based on Tesauro's seminal success with TD-Gammon in 1994, many successful agents use temporal difference learning today.[3] But in order to be successful with temporal difference learning on game tasks, often a careful selection of features and a large number of training games is necessary. For board games of moderate complexity like Connect-4, we found in previous work that there are required a very rich initial feature set and several millions of game plays. So they work with different approach for speeding up for such a complex task. Even they also proposed a new variant of TCL with geometric step size changes.

A Shannon C-type strategy program, VICTOR, was used to easy the Connect4 game with AI that at least draw was happened in the game. Using the combination of conspiracy-number search, search tables and depth-first-search, here build an opportunity to show that one can win on the standard 7*6 board. Using a database of approximately half a million positions, VICTOR can play real time against opponents on the 7 × 6 board, always winning with one.[1]

This paper mainly introduced a self-developed heuristics supported by well-demonstrated result from researches and our own experiences which fighting against the available version of Connect-4 system online.[2] Previously Connect4 was built with traditional zero-sum approach which was

not being popular and less effective. So this method will be effective for Connect4 and this research is also going on now a days. Still it does not play with super optimal way. Von Neumann and Morgenstern proposed theories of rational choice for a special class of situations in which a person is faced with choosing between various alternatives, and where the person knows that the outcome of his choice is in part a function of how another person, who is presumed to be equally rational, chooses.[2]

3. Motivation:

Having various field, we have found that board game is a one of the interesting field of AI. So we got interest in Connect4. Also it is a childhood game of ours. It memories us about our childhood. By developing this game in human v/s device, we can also learn about artificial intelligence and play the game with our pleasure. It helps us to learn about the board game logic of AI and also helps us learn about the acting of AI.

4. Objectives:

Every project activity has a specific goal to perform. Here, in this project the intention was to make the game difficult with AI and also having some probability of winning human. But main focus is about winning probability of AI and we have tried to implement it. We have focused on the following things:

- Human v/s human game plot
- Human v/s AI game plot
- Level of AI

5. Limitations:

In our project, we have some limitations. These are given below:

- Our game isn’t android application based
- It will only run in project environment
- Game is not too fast
- Board size of the game is fixed(6*7)

Chapter B

1. Literature review:

In this work they investigate different approaches of online-adaptable learning rates like Incremental Delta Bar Delta (IDBD) or temporal coherence learning (TCL) whether they have the potential to speed up learning for such a complex task. They propose a new variant of TCL with geometric step size changes.[3] They compare those algorithms with several other state-of-the-art learning rate adaptation algorithms and perform a case study on the sensitivity with respect to their Meta parameters. They show that in this set of learning algorithms those with geometric step size changes outperform those other algorithms with constant step size changes. Algorithms with nonlinear output functions are slightly better than linear ones. Algorithms with geometric step size changes learn faster by a factor of 4 as compared to previously published results on the task Connect-4. Here self-tuning learning algorithms like IDBD, which automatically adapt certain hyper parameters as learning progresses.[3] This avoids manual intervention and there is hope that the same self-tuning scheme can be applied to other games as well and that the insights gained are transferable to other learning tasks as well.

Heuristic function is used in Minimax for evaluation of the current situation of the game. The final decision made by Minimax largely depends on how well the heuristic function is. Therefore, designing a reasonable heuristic function is paramount. In this research, they designed a heuristic function for Connect-4. To evaluate the current situation of the game, the heuristic function firstly looks for different features on the board and then gives them proper values. Finally, the heuristic function returns a summation of all the values of features on the chess board.[2]

VICTOR is an implementation of these rules, achieving one of the goals of the project: a Shannon C-type strategy program to play Connect-Four.[1] There has more trouble with positions where both players seem to have chances. Often it is detected so fast that one player is winning too fast. Then the more difficult positions are decided by tactical threats, which has to be solved by

searching instead of evaluating. Here strategic rules had been defined, those rules are mostly detected by expert player of Connect4 game. This algorithm is evaluated for any board. When the game-theoretical value of a position had been determined, the search table was built to a file. It is happened with every gaming mood. And after calculating the table next move is done by program.

Chapter C

1. Methodology:

Connect-4 game is a chess game on a board of 7 vertical columns of 6 squares each. Two players make their moves in turn till 4 men are connected horizontally, vertically or diagonally. Once a man is put in one of the columns, it will fall down to the lowest unoccupied square in the column. In our game, we designed an intelligent program to seek for the best move, using Minimax.

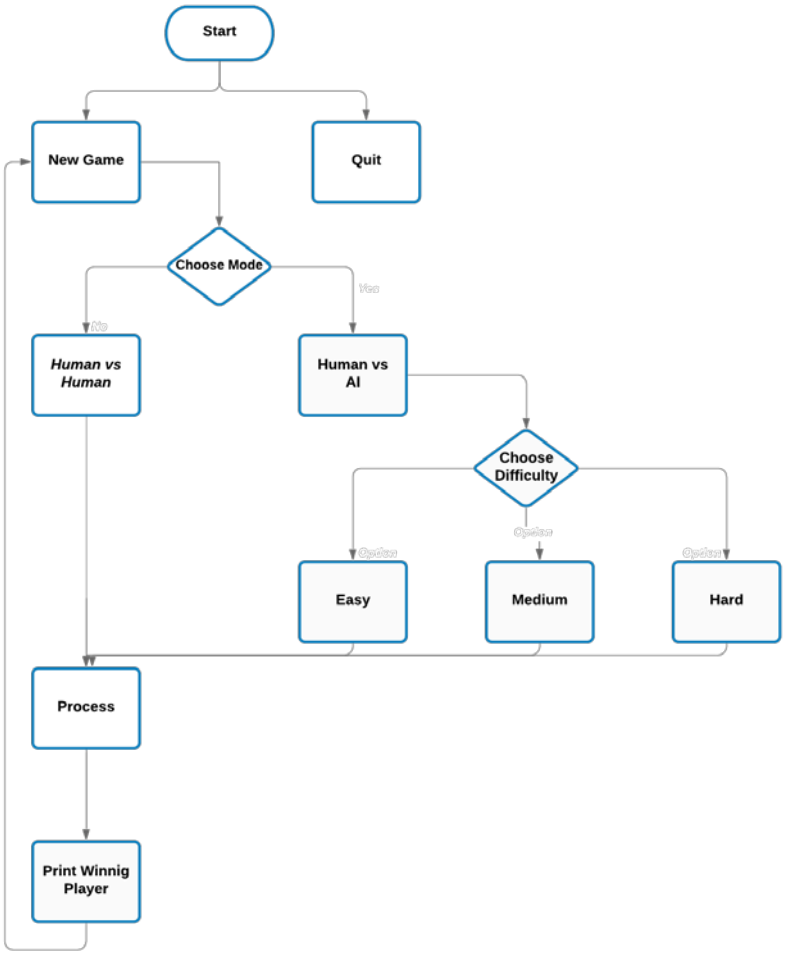


Figure1: Flowchart of Connect-4 game

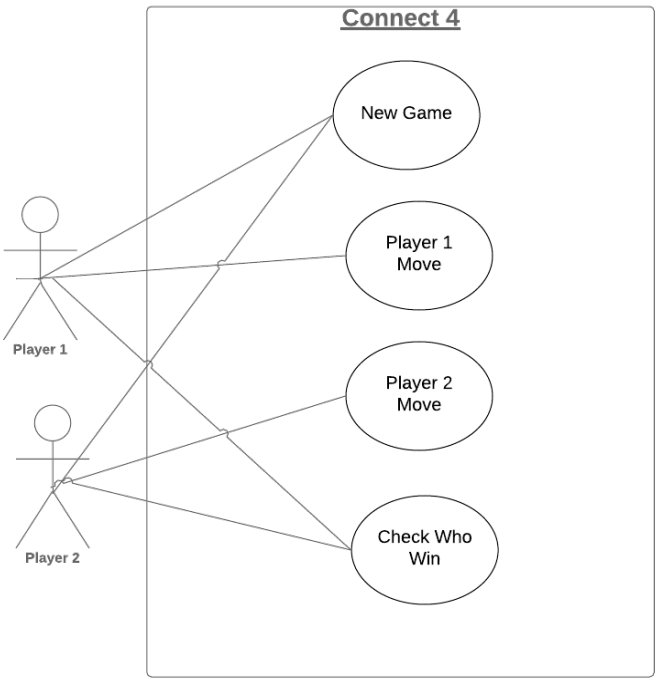


Figure2: Use-Case Diagram of Connect-4

1. In this game first we create a board of 6x7 that's mean 6 row and 7 column in the board. Also in our interface there are two buttons start game and the other is quit game.

2. After click the start game button player 1 needs to choose the game mode, he/she wants to play with another human player or with Artificial Intelligence that we created in computer mood.

3. In Human v/s Human two human can play and after the end of the game it shows who wins.

4. In Human v/s AI there are 3 different levels Easy, Medium and Hard.

5. In Easy difficulty depth of Minimax tree is only 1 so it's very easy to win but in Medium level depth of game tree is only 3 so it is quite hard to win but not impossible. In hard level depth of game tree is only 7 so we can say that AI beat Human maximum times also can guess the probability is above 60 percent.

Chapter D

1. Implementation:

In our game we used the Minimax algorithm for creating computer mood an artificial intelligence that can beat the Human all the time.

Minimax is used in artificial intelligence for decision making. In most cases, it is applied in turn-based two player games such as Tic-Tac-Toe, chess, etc. In our Connect-4 game, Minimax aims to find the optimal move for a player, assuming that the opponent also plays optimally.

In Minimax, there are two players called Max and Min. Starting with Max trying its first move, Minimax algorithm will try all the possibilities of combination of Max's and Min's move. When either one wins or the game comes to a draw, an evaluation value of the board will be given to indicate the situation of the board. If some features on the board are in favor of Max, a positive value will be given to that feature. Otherwise, a negative value will be given. The final evaluation value is the summation of all the values of features. Max will choose the maximum evaluation value and Min will choose the otherwise. Eventually, Max will decide the best move. A descriptive figure is shown below.

```
function minimax(node, depth, maximizingPlayer) is
    if depth = 0 or node is a terminal node then
        return the heuristic value of node
    if maximizingPlayer then
        value := −∞
        for each child of node do
            value := max(value, minimax(child, depth − 1, FALSE))
        return value
    else (* minimizing player *)
        value := +∞
        for each child of node do
            value := min(value, minimax(child, depth − 1, TRUE))
        return value
```

Figure3: Pseudo-Code of Minmax

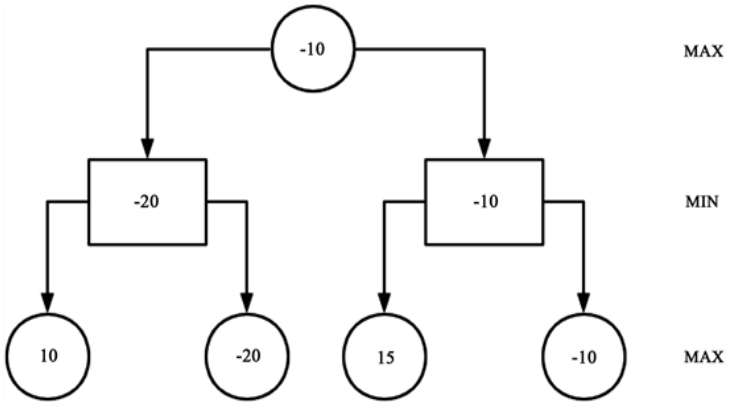


Figure4: A possible game tree

Figure 4 is a possible game tree. When it is Max's turn, he has two possible moves. It can either result in the left situation or the right situation in the second layer of Figure 4. Then, it is Min's turn. Under each of the two situations, Min also has two possible moves. And it will result in four possible situations which are terminal nodes that indicate the search stops because of either the game ends or the maximum search depth is reached.

Then, the evaluation value will be given according to the features of the situation. The numbers in the circles of the third layer are the final evaluation value. Min in the second layer will choose the minimum number from its children. Therefore, −20 and −10 are chosen. Then, Max in the first layer will choose the maximum number from its children. Therefore, −10 is chosen. Eventually, Max will decide that the right path is the best choice.

The flowchart for Minimax algorithm is illustrated as following Figure 5.

In our game, this algorithm is specifically for Connect-4 chess game. To decide which column to play, it creates a game tree by trying out all the possibilities of combination of Max's and Min's moves. Each node of the game tree is initially given a heuristic value. If it is Max node, it'll give minus infinity. If it is Min node, it'll give infinity. When the pre-set depth is reached or the game ends, the heuristic value will be returned to its father node. The father node compares the returned value and its value and chooses the bigger one. When the root node receives a returned heuristic value, it will record not only the bigger heuristic value but also the column number corresponding to the value. After the root node tries out all the seven columns, the best column is recorded. Hence, the algorithm finds out the best move.

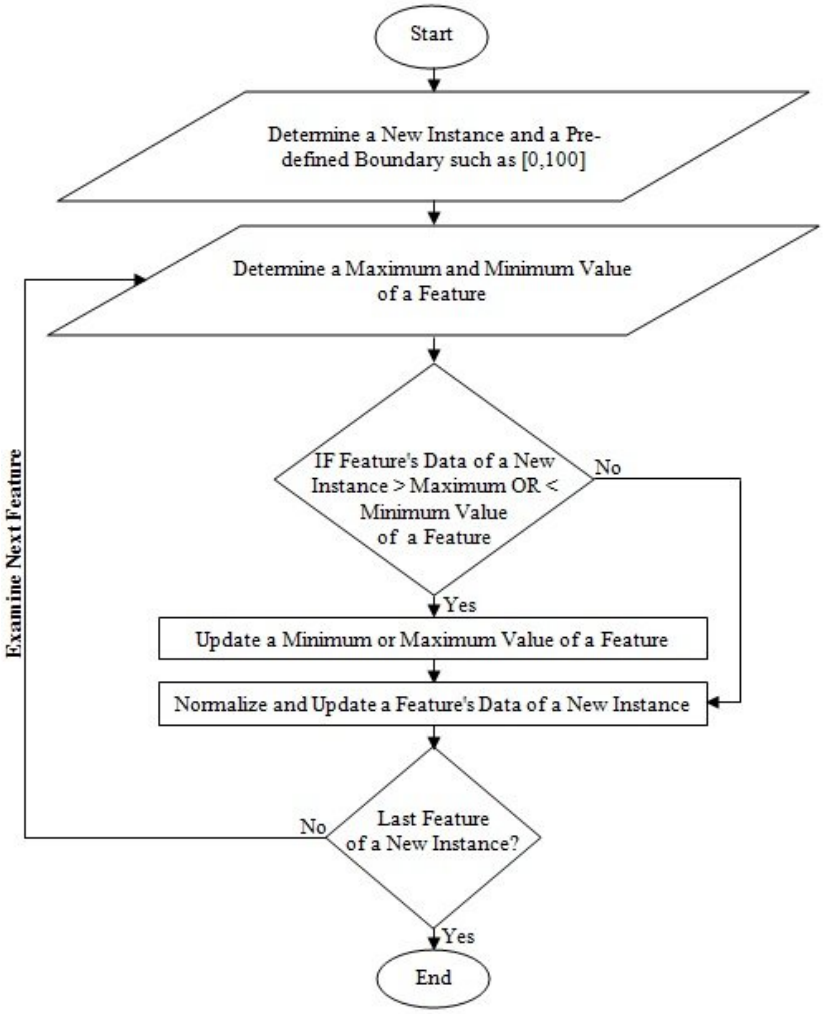


Figure 5: Flow Chart of Min-Max Algorithm

Platform for this game: Sublime Text Editor, Anaconda

Language: Python

Package needed: pygame, NumPy, math

PC Configuration: Any PC that could be installed Linux, Windows or other operating system. Also a dual core Pentium 4 or AMD A4/6 processor with minimum 2GB of RAM would be enough.

Chapter E

1. Result and Output:

Figure 6: Start Menu

Figure 7: Choose Mode Menu

Figure 8: Choosing Difficulty

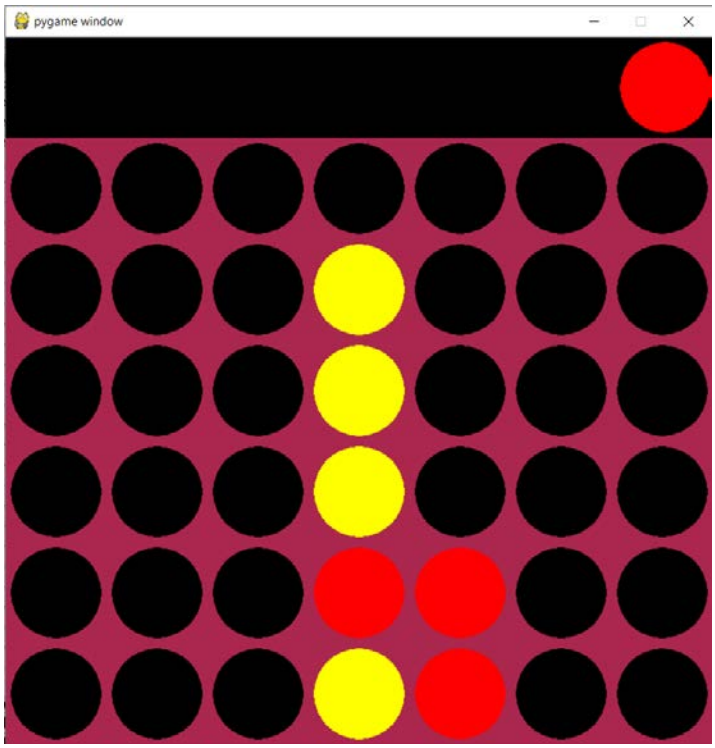


Figure 9: After starting game

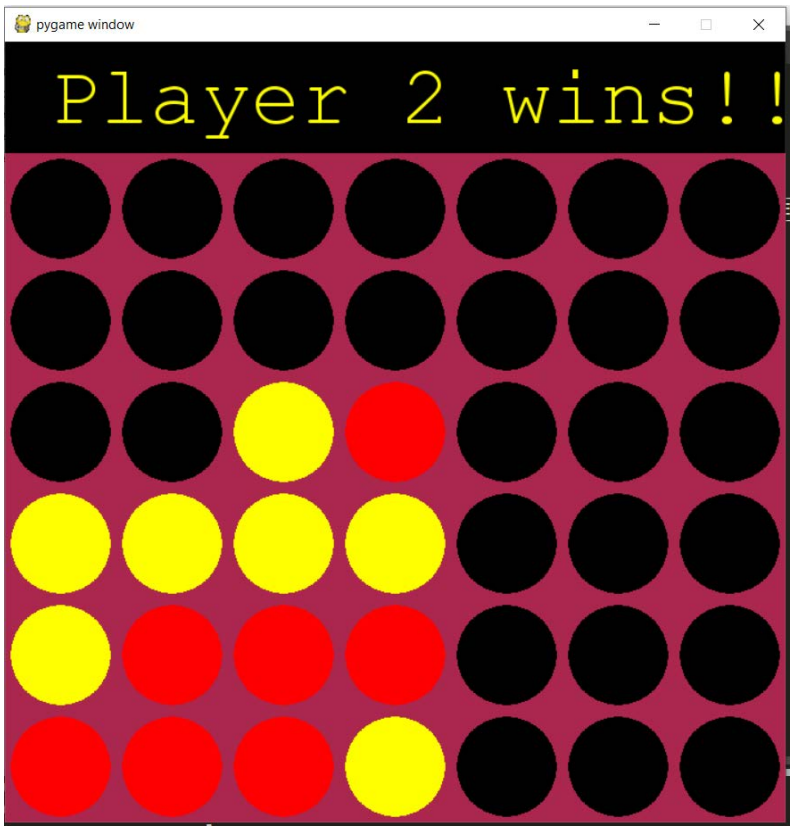


Figure 10: Winning state

Chapter F

1. Conclusion:

In this project, we have learnt many thing of AI and gather a vast knowledge of AI though we have some limitations. In future we will try to solve our limitations. Now a days, AI is used in every era of our life. So, it is important to learn AI. And python is a high level language. It is also important to learn this language because it makes all types of programming very easy. From this project, we learn both AI and python.

Chapter G

1. Reference:

[1]. http://citeseerx.ist.psu.edu/viewdoc/versions?doi=10.1.1.38.2778

[2]. https://www.scirp.org/html/1-9601415_90972.htm#

[3]. https://ieeexplore.ieee.org/abstract/document/6945857/citations#citations