The Queen's Gambit

Week 1 Tasks

Abhilasha Sharma Suman

1 Chomp

Q: Does there always exist a sequence of moves such that the player making the first move wins?

- Yes, there does exist a winning strategy for the player that moves first but there is no definite sequence of moves which will lead to the said player winning in a general m*n sized chomp board.

  For the prediction of the winner from a random position, we can visualise this as a composition of sub rectangular configuration. Depending on the configuration and the optimal move that the player can make and the position they can force their opponent into, we should be able to predict the winner of the game from a specific position.

2 The game of Nim

Q: Does there always exist a sequence of moves such that the player making the first move wins?

- No, it isn't always that a sequence guaranteeing the first player's win exists. If we were to represent the number of sticks in each heap in binary system, and perform XOR function on these and get the result, say x. If at starting position, x is zero, then the first moving player doesn't have a winning strategy given that the opponent also plays optimally. For non zero x, the first player has a winning strategy, irrespective of how the second player moves(optimally or not), given that he/she plays optimally.

Q: Given a Nim position, is there a way to determine whether it is possible to win for any player with perfect play? If not, why? If yes, how?

- Yes, we can always determine if it is possible for a player to win at a given position in a Nim game. This is because each position is like a new game of Nim with the player we are talking about being the first player in the position. If we calculate the xor of the number of sticks present in each heap, say a. If a is zero then the player whose turn it is will not have a winning strategy given that the opponent plays optimally. If a in non-zero then the player whose turn it is, has a winning strategy irrespective of how the second player plays(optimally or not), given that he/she plays optimally. This quantity x is the Nim-sum of the game.

Task: If a sequence of move which guarantees a win does exist for a Nim position, how will you determine the sequence of moves? That is to say, how is the computer coded to play the most optimum way?

- The approach I took to this problem was this: the computer should be coded to try and leave a zero nim-sum situation for the opponent. This way every time the second player plays, it will be as if he/she is the first player in a game of nim where the initial state of the game is such that the nim-sum is zero putting the opponent in a losing position despite optimal play from their side. In case the nim-sum of the initial state (for the computer), is zero it will play randomly and hope that the opponent makes a mistake and the nim-sum turns out to be non zero when it is the computer's turn. The link to this code implementation is here: Nim.py

3 Greedy or Not

Task: You have to write a code which takes input a number n which is the size of the list and then the element of the list. It then outputs which Player wins if both play optimally.

- The approach I took to this task was to create a function that I created a mapping array. I took the sub-arrays that would be formed based on different choices that the players make and then filled the mapping array with the highest score difference a player can possibly create with his/her choice. Then finally, if the maximum difference of score that player1 can make is positive, then player1 will win given that he/she plays optimally, if it is negative then player2 will win otherwise it will result in a draw. Here is the code implementation of this logic: GON.py