Tec de Monterrey Campus Querétaro

Arturo Rivera Flores - A01202457
Cuauhtémoc Suarez - A01206503
Francisco Villanueva - A01202413

Artificial Intelligence
08/23/16

# Report Connect 4 with Digit Recognition

We are a team of three persons that aimed to develop an excellent project related to artificial intelligence.Two of our three members are studying a robotics major. So we decided to create a real connect 4 game.
This part of the deliverable will only explain the algorithm that the computer will use in order to win the game. Using AI is the best way to solve this problem, because if you use a random variable to choose where to insert the next coin you could probably will loose all the games and it will be boring for the person that is plating against the computer.

The algorithm that we used for the connect 4 is the min max algorithm. At the beginning we search for a video on youtube(1) that can explain the algorithm to us in a very graphical way. Then we searched the min max algorithm in games pages.

Min max algorithm is the best way to solve our problem, because this algorithm is applied in two player logic games, like chess, tic-tac-toe and connect 4 . This algorithm will follow some rules and premises, with that information it will give a valid move . Another important detail is that min max algorithm is only for full information games.

This algorithm required a search tree and a depth. The way min max works is :

1. Your initial state is your root state. The depth, tells how many levels the machine would go through.
2. Make a fixed Breadth-first search till the end or a final state is found (Losing, winning or there are no moves left).
3. You have an heuristic.
    a. The first heuristic we implemented for the project was: Winning return 1000 points, lossing return -1000 other case return 0. The team realize that this was not retrieving the most inefficient way. So we change it to : if you have 1 coin alone return 1, if you have 2 coins together return 2, f you have 3 coins together return 3. We based our heuristic on this page https://www.gimu.org/connect-four-js/jQuery/alphabeta/index.html
4. Then you will have a tree like (Image 1)

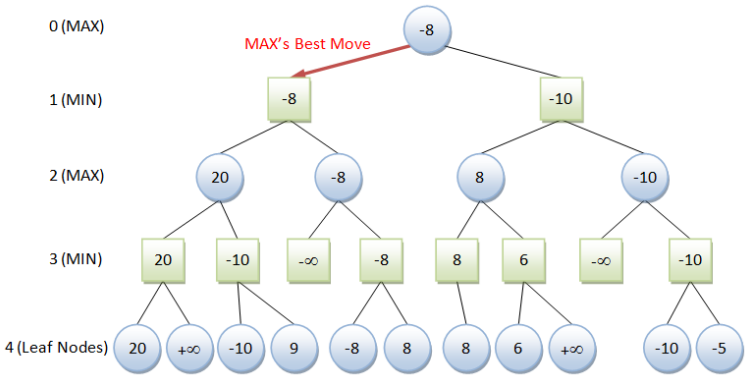


Image 1

Circle is your turn, so you obtain the biggest number. Squares are your opponents turns so you obtain the lowest number. In Image 1 we will explain you start in the lowest level so you.
If it your opponent's turn you will return the lowest number but if it is your turn you can return the biggest number.

From the leaf node to the third level you will return the lowest number because is your opponent's turn. Between 20 and infinity the lowest is 20, then for the next: between -10 and 9 the lowest number is -10 and so on. Then from the level 3 to the level 2, the algorithm will search the biggest number for example: between 20 and -10 you choose 20, between minus infinity and -8 you choose -8 because is the biggest and so on then you repeat the same steps from the second level to the first level. Thus, at the end it retrieves the best option.

These values helps to show how good a game move is. This will make you make maximize your points but also will not allow your opponent to win that easily.
**Important note: The deepest you search the efficient solution you get but also is more expensive.**

Then we have a problem, the algorithm took a lot of time to perform a good solution. So we decided to improve the algorithm with another technique called the alpha beta pruning. This algorithm helps the min max algorithm by cutting of the decision tree according to alpha and beta . Every node has an alpha and beta. (Image 2)

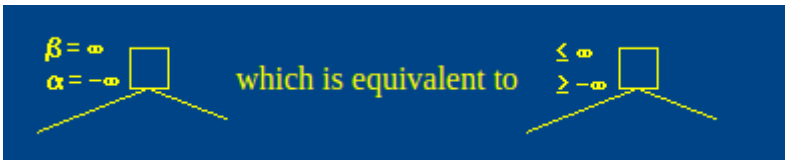


Image 2

http://web.cs.ucla.edu/~rosen/161/notes/alphabeta.html
beta = minimum upper bound of possible solutions.
Alpha = the maximum lower bound of possible solutions.

Alpha beta pruning will start in the root node and then do a fixed Depth First search until you reach a final state or reach the depth. (Image 3)

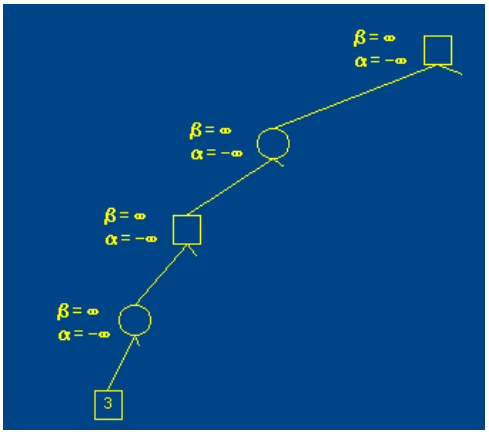

Image 3 Use a Depth First search

That value becomes your beta, and then go to the next node. If that value is lower than your actual beta then it becomes the new beta. (Image 4)

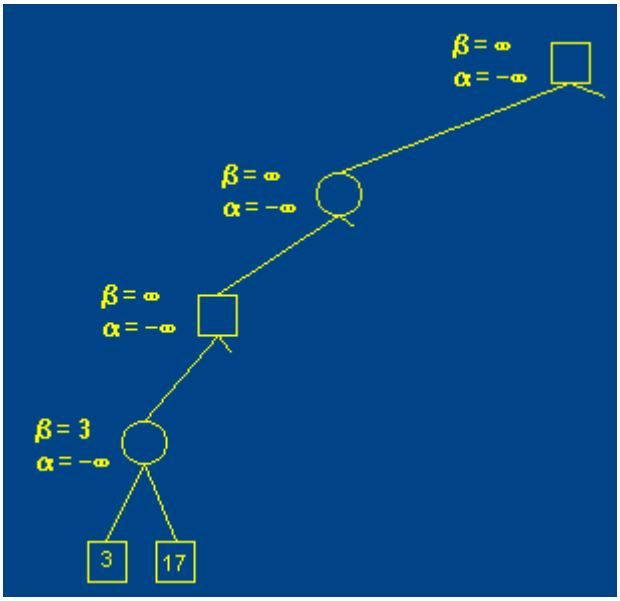

Image 4 Beta is used in circles and alpha in squares

Then the parent node alpha's becomes equal to their child beta, then it go to their other child, has the same alpha and beta, and explore the next node (Image 5)

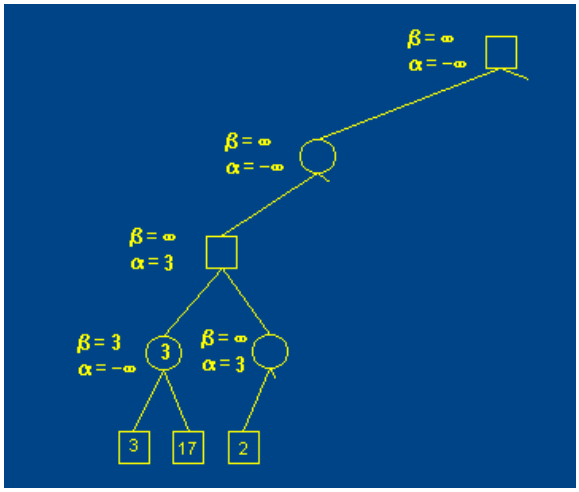

Image 5 the child node will share the same parameter.

In this min node the value of this node will need to be (2>=) it changes the beta to 2. Now there is no overlapping between the regions bounded by the alpha and beta. The algorithm will find a value that is not only greater than 3 but also less than 2. This is impossible, so it will stop searching in that children. (Image 6)

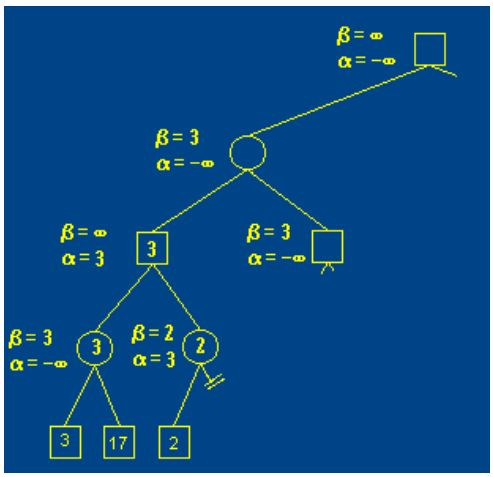

Image 6.

And so on. Therefore, you will get a tree like this.

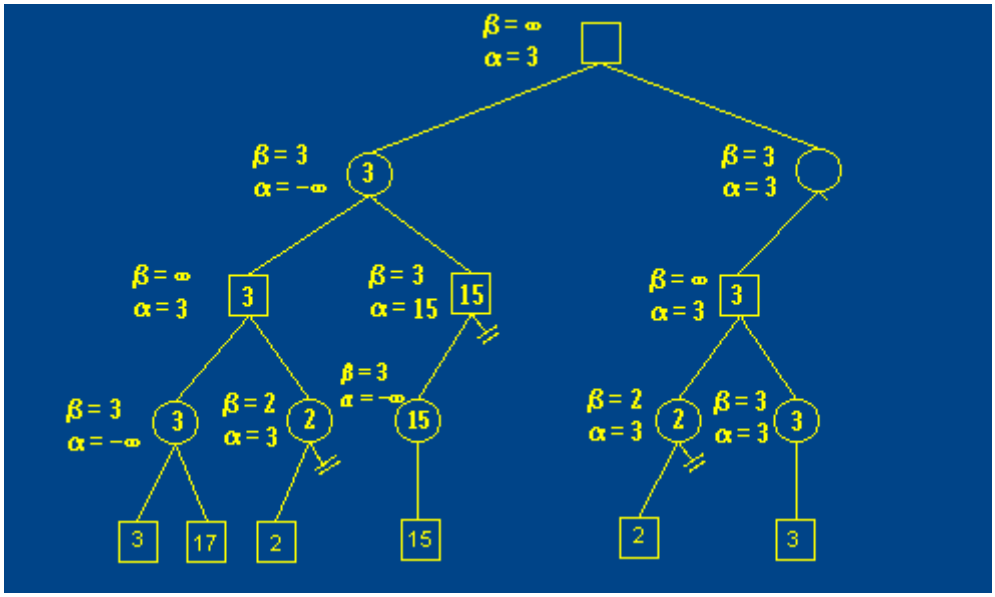


Image 7.

We investigate another way to implement the connect 4 game. We found that it can be developed using neural networks, we discussed and concluded that it would need a lot of training and that it would not find efficient moves.

In order to train the neural network we needed a dataset of inputs and a correct output. The inputs would be the matrix with the state of the connect four and the output will be the best move. These neural nets can be configured in many different ways, so we would need to make many upgrades to obtain better results.

Github repository:

https://github.com/franzmau/Connect4

References:

1. https://www.youtube.com/watch?v=6ELUvkSkCts&t=2s

2 A. (2002, July 28). Introducing the Min-Max Algorithm. Retrieved November 21, 2016, from http://www.progtools.org/games/tutorials/ai_contest/minmax_contest.pdf

Flying machine (2010 January 10) An Exhaustive Explanation of Minimax, a Staple AI Algorithm http://www.flyingmachinestudios.com/programming/minimax/

3M,Thill.(2012),Reinforcement Learning with N-tuples on the Game Connect-4 Retrieved November 21, 2016 from http://www.gm.fh-koeln.de/~konen/Publikationen/ppsn2012_RL-CFour.pdf