## <https://youtu.be/wD_WzaXKb_c>

# **Dots and Boxes with Weighted Scoring**

## Overview

Dots and Boxes is a classic pencil-and-paper game. In this assignment, the traditional game is modified to include weighted scoring, adding an additional layer of strategy. The game is implemented using Scala, offering an engaging experience for players.

## Scope of the Assignment

* **Weighted Scoring**: Each box is assigned a random value between 1 and 5, and a player's score is the sum of the values of the claimed boxes.
* **AI with MINIMAX Algorithm**: The AI opponent utilizes the MINIMAX algorithm to make strategic moves. The human player can set parameters such as the number of plies the AI will search and the size of the board.

## Implementation in Scala

* **Functionality**: The Scala implementation ensures the game works as intended with a user-friendly interface.
* **Algorithm Understanding**: The code demonstrates a deep understanding of the MINIMAX algorithm, ensuring efficiency and cohesion.

## Pitfalls and Considerations

* **Performance Challenges**: Experimentation with different plies and board sizes may reveal performance challenges, especially for larger boards and higher plies. The analysis document delves into the reasons behind these challenges.

## Analysis

For a detailed analysis of the implementation, including insights into gameplay experimentation, performance considerations, and reflections on AI behavior, refer to the Analysis Document.

## **Performance Challenges**

Experimenting with the plies and board sizes quickly became too much for my computer. Maybe that’s because it’s outdated, or my implementation is unsatisfactory. I choose the latter. This is important because I thought it could be able to calculate it quickly for 2x2 boards, but it took considerable time. It seems the plies and depth limit, even with some optimization, will always be costly, which minimax is inefficient with lots of game states and leaves to uncover. This is why we saw the MCTS search would be efficient, as it could go down the tree much faster. Although it would have to simulate many games to generate the best “path,” it could do it within a certain degree and quicker.

## Move Analysis on 4x4 Board

When I was looking for the 4x4 board with a three-ply, the first couple of moves always took a couple of minutes, which was scary, but I knew it would be after playing against the AI for the next couple of moves beating me with the moves I made. This made sense because it calculated what moves I could do with the higher ply limit set, which meant it could search down its tree and see which moves it could make that would maximize its utility while I chose the maximum utility.

## Optimal Ply Selection

For the question regarding the number of plies that would make it too easy, challenging, and challenging, I saw that ply from 1-3 would be too easy, and the AI would continually make mistakes or let me complete a box. This meant the AI was not searching far enough as specified and didn’t realize it would be screwing itself over with such a low ply limit. A good experience would be the sweet spot of 5 plies with a 3x3 board or 2x2 as it calculated its moves much better, and my move would be blocked much easier. When we reached plies of >5, we saw the AI improve its game and play, which beat me many times. It seems that the plies that are too large will not allow humans to win. It becomes impossible until humans develop the ability to calculate a minimax implementation of the dots and boxes game.

## Board Size Considerations

In particular, I saw that board sizes greater than 5x5 with a ply limit of 4 would be doomed to fail. Other than the 25 boxes that can be claimed, it also needed to calculate the giant spaces and plays it can move based on the available moves and different alpha-beta values it had to calculate to percolate the tree.

## Tradeoff: Challenge vs. Time

I think there is a tradeoff between the challenging experience and the time it takes for the AI to make a move because the more time it takes to make a move, the less suspenseful it is, and although the AI may win, the human has to wait. Games that are quick and easy to start once a game, like Tic-Tac-Toe, are fun because of how quickly the game is played, not who wins the most times. Of course, there’s a strategy to play, but if you win every time that takes too long for another game to be started and finished, where’s the fun in playing a game with an AI? In other words, once the human has to wait for more than the average time it takes for someone to listen to an entire 3-minute radio song, the game becomes boring and stale. There is no rush to choose a move because AI will always take longer than you.