Connect 4 – Alpha Beta writeup

1. My heuristic consisted of adding up all the different tokens within a space of 4 spaces. It used a subsection of each row, column and diagonal with 4 spaces in it, and gave a higher number to those spaces that had more of the same player token. It then did the same for the opponent, so each state consisted of the value of the state for the player minus the value of the state for the opponent. This way, not only did it favor its own success, but it favored preventing the opposing player from getting in a good state. I used this one because I found that it was an easy way to quantify the different states of the board.
2. With 10 seconds, I had no problem exploring all 4 depths of the tree (this was my initial setting). One thing I did notice was that every once in a while I would get a DPS that would take a little longer than 10 seconds, so even though normally my algorithm would function fine with 10 seconds, it still isn't as optimal that it could be in timed constraints. With 3 seconds, I couldn't get through the whole tree. I would normally be able to search around 3 branches up to depth 4 within that limit before timing out. However, if I move the depth down to 3, I can get all the way to the last branch before timing out. With 5 seconds, I can get all the branches to a depth 3, but with a depth of 4 I can get to the 5^{th} branch. So, there is not much improvement between 5 and 3 seconds with a depth of 4. It is interesting how much quicker the algorithm runs just by limiting the depth by 1.
3. So far, I have yet to beat my algorithm. I even challenged my wife and friends to play it, and it doesn't matter whether it goes first or second, it always seems to either end up winning or in a tie. I have tried doing random moves, as well as looking up good moves (obviously I didn't follow these the whole game). It does a very good job at setting itself up for future moves.
4. When the ai plays itself, I find it really fascinating because for a long time it seems to mirror itself. At the beginning, there was a slight difference when one player blocks another, but then a few turns later it went back into the same pattern that it followed before. I've run the algorithm a few times and it seems that it always ends up like this. The first player always seems to end up winning. This behavior is expected because the ai will always react to the state before with the same response, so even if they played each other over and over again, we would get the same result.