Rishabh Shrestha

Implementation of AI players

I implemented two different AI players for the game-Rubik's tile. AI-1 is Ace and A1-2 is Bob. A1-1 and A1-2 are fairly different in terms of their move and the way they read the game.

I implemented the smart players using the Minimax algorithm with alpha-beta pruning. Both of them go to maxDepth of 2. The max function is called before AI takes any decision to flip or move. The max function returns values such as:

1. Row and column of tile to be flipped
2. The new row and column of the tile to be flipped
3. The row, column of where you want to place your move
4. The color of tile to be faced in your move
5. The value from evaluation function

The row and column values returned are used to flip and place the tiles
The value from the evaluation function is the value used to decide the best possible moves.

The max function takes as input the current state and a depth measure including alpha and beta value. The initial values of alpha and beta that are passed to max function are -1000 and 1000 respectively. In the minimax algorithm, the maximizer tries to get the highest score possible while the minimizer tries to do the opposite and get the lowest score possible. In the end max function returns the max possible score given the state of board and maximum permitted search depth.
Here is what max does:

1. If we have reached the max possible depth allowed or if the state of the board is terminal then return the current value of the state.
2. It uses the combination of each possible flippable tile of the opponent, all possible valid flips for each flippable tile, possible row, column pair to place a new tile, two colors of the tile. At first it flips, and then, for every possible row, column pair for placing the new tile, it checks if the move is valid and then tries both of the possible colors.
3. After flipping and making the move, it calls min to find how good the move is.
4. If min says it is better than the best move so far then it flags the move as best move.
5. And for alpha beta pruning, if alpha is greater than beta cutoff, return it

Evaluation function is used to score the moves. Ace and Bob have a little different evaluation functions. Ace checks for winning position and gives it a score of 10 and -10 for losing position. It checks for all possible winning positions-row, column and diagonal and 3 tiles involved in winning positions need to be locked too. It also takes a score of 7 for draw which means it is inclined towards taking draw when the situation comes

However, Bob would not settle much for a draw. It would just take a score of 2 if there is a draw. In addition to checking for winning, losing and draw cases, it also checks if there are 2 in a row of his tiles in locked position and takes a score of 3 and if the opponent has the same condition then it takes -3.

Sample Games:

The 100 games between Random and Ace(AI-1) ended with following wins:

Random: 4
Ace: 89
TIE: 7

The 100 games between Random and Bob(AI-2) ended with following wins:

Random: 2
Bob: 94
TIE: 4