MinMax

No projeto, existem 12 classes: Heuristica, Ia, Jogada, MinMax, MonteCarlo, No, Opcoes, Tabuleiro, TelaCarregamento, TelaFinal, TelaInicial e Vez.

- TelaInicial: é a primeira tela carregada ao iniciar a aplicação. Ela contém os botões "Multiplayer" e "IA" para o usuário escolher como deseja jogar.
- Opcoes: se o botão "IA" for selecionado, um painel será aberto para o usuário selecionar as características da IA, sendo elas: qual tipo de inteligência deseja enfrentar, podendo ser MinMax, Monte Carlo e Heurística. Se a opção selecionada for MinMax, também será possível selecionar o nível da IA, que varia de 1 até 10.
- TelaCarregamento: é a tela exibida após a primeira jogada do usuário caso o modo selecionado seja MinMax ou Heurística.
- TelaFinal: é o menu aberto ao acabar um jogo, onde é possível escolher jogar novamente com as mesmas configurações da partida anterior ou retornar para a tela inicial.
- Vez: enum que contém os valores "IA" e "JOGADOR".
- No: é a classe que contém a estrutura dos nós da árvore. Possui uma lista para guardar as jogadas que ainda são possíveis a partir do nó atual, um inteiro para guardar o valor de MinMax, um enum Vez e um vetor para os filhos do nó atual.
- Ia: é a interface que contém os métodos "jogada", "jogadaInicial" e "finalizar".
- Jogada: contém as características de uma jogada, como os vizinhos do quadrado selecionado, o quadrado atual e os quadrados que podem ser coloridos caso seja feito um ponto nessa jogada.
- Tabuleiro: é a classe que constrói visualmente o tabuleiro, além de realizar o tratamento das jogadas. Nela contém um vetor do objeto Jogada de tamanho 12, um para cada jogada possível, contendo todas as especificações da jogada. Cada casa do tabuleiro possui um listener, onde, quando o jogador clica em uma casa, ela é marcada e os devidos tratamentos são feitos. Na

classe, existe também a função "fazerJogada", que realiza o tratamento tanto quando a IA faz uma jogada quanto quando o jogador faz uma jogada. Essa função recebe um inteiro que representa qual posição deseja ser jogada, além de receber um boolean indicando se a jogada foi feita pela IA ou pelo jogador.

- MinMax: é a classe que possui a implementação de uma IA pelo método minmax. Nela contém as seguintes funções:
    - "jogadaInicial()": faz a jogada inicial do jogo, escolhendo aleatoriamente uma das jogadas possíveis. É uma função implementada pela interface IA. Complexidade: Θ(c), c = constante.
    - "finalizar()": reinicializa o objeto raiz e atualiza o valor da variável dificuldade. É uma função implementada pela interface IA. Complexidade: Θ(c).
    - "percorrerArvore(int jogada)": navega na árvore de possibilidades do jogo até o nó correspondente à jogada informada por parâmetro. Complexidade: O(n), n = número de nós filhos do nó atual.
    - “jogada()”: realiza uma jogada no jogo, escolhendo a melhor opção dentre as possíveis, levando em consideração a dificuldade do jogo. Essa função é implementada pela interface Ia. Complexidade: Θ(n).
    - “construirArvore(No n)”: constrói a árvore de possibilidades do jogo, partindo do nó informado como parâmetro. Complexidade: Θ(j!), j = número de jogadas, no caso j = 10.
- Heuristica: Classe muito similar à "MinMax", com a diferença que o método de avaliação dos nós da árvore é diferente. As folhas da árvore recebem o valor 1 quando levam a uma situação de vitória para a IA, 0 em caso de empate e -1 quando o jogador alcança a vitória. Para os nós que não são folhas, primeiro é verificado de quem é a vez naquele momento. Se for a vez da IA, serão desejados filhos com minmax positivo. Caso contrário, negativo. Para cada nó, são percorridos todos os seus filhos e é somado em uma variável os valores negativos de minmax e em outra variável os valores

positivos de minmax. Depois disso, é verificado se a variável desejada, ou seja, a variável com os valores negativos ou positivos dependendo de quem é a vez, não é zero. Se essa variável desejada não for zero, então seu valor é retornado pela função. Se for zero, é retornado zero caso exista algum nó com minmax zero. Se não existir, é retornado o valor da outra variável. A escolha da jogada aqui é baseada no nó filho que possui o maior valor de minmax.

- MonteCarlo: implementa uma IA que simula jogos aleatórios para basear sua tomada de decisão. Para cada jogada possível no momento, é simulado um número de jogos aleatórios após a jogada em questão ser feita. O número de jogos é definido pela variável "QUANTIDADE". Ao final de toda a simulação, é escolhida a jogada em que suas simulações levaram ao maior número de vitórias em comparação com as outras. A complexidade de cada jogada da IA é dada por:

  Complexidade: $\Theta(\sum_{i=1}^{j}(i * q))$, j = número de jogadas possíveis no momento, q = quantidade de simulações.