DEEPAK GALA (800584049)

AI HW 4

1. **DESCRIPTION OF THE SET OF STATE**

   The state of the game at any point is the vertices in the graph with associated colors assigned. For example, for the graph below {[1, Red), (2, Blue), (3, Uncolored), (4, Uncolored), (5, Uncolored)} is one possible state, where each tuple inside the set is given by a number representing the vertex and an associated color besides the vertex number.

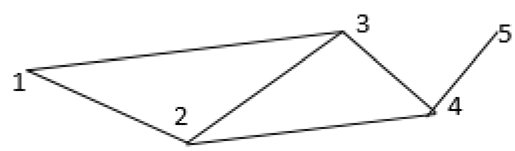

Figure 1: Graph A

So, the set of state will be all possible such states.

Examples:

1. Initial state:
   S_0 = {[1, Uncolored), (2, Uncolored), (3, Uncolored), (4, Uncolored), (5, Uncolored)}

2. One of the Terminal state would be:
   S_{T1} = {[1, Red), (2, Blue), (3, Red), (4, Uncolored), (5, Uncolored)}.
   A loses in this case.

   S_{T2} = {[1, Red), (2, Blue), (3, Uncolored), (4, Blue), (5, Red)}.
   B loses in this case.

3. Couple of states while the game is ON:
   S_X = {[1, Red), (2, Blue), (3, Uncolored), (4, Uncolored), (5, Uncolored)}
   S_Y = {[1, Red), (2, Blue), (3, Uncolored), (4, Uncolored), (5, Red)}.
   The game will continue.

2. **PLAYER FUNCTION**

   Given the current state, the player function decides of the two players whose turn it is to play. This can be achieved by simply counting the number of Red and Blue colors in the state.

   a) If the number of Red color = number of Blue color ➔ It will be A's turn
   Example: player(S_0) will return **A** player(S_X) will return **A**

   b) If the number of Red color > number of Blue color ➔ It will be B's turn

Example: player(S_Y) will return **B**

Note that number of Red color will never be less than number of Blue color.

3. **RESULT FUNCTION**

Given the current state **S** which is not a terminal state (terminal (S) = False) and action **A** with a player's turn player(S), the result function will define the corresponding successor state **S'**.

For example:

1. Since terminal (S_0) = False and player(S_0) = A, the output of the result function for an action {(1, Red)} will be:

   result (S_0, {(1, Red)}) = {(1, Red), (2, Uncolored), (3, Uncolored), (4, Uncolored), (5, Uncolored)}

2. result (S_X, {(5, Red)}) = {[1, Red), (2, Blue), (3, Uncolored), (4, Uncolored), (5, Red)} = S_Y
   result (S_Y, {(3, Red)}) = {[1, Red), (2, Blue), (3, Red), (4, Uncolored), (5, Red)}. = S_{T1}

3. result function for a terminal state S_{T1} and S_{T2} cannot be called or will return None as terminal (S_T) will be True.

4. **TERMINATION TEST**

The objective of the termination test is to determine if the game is over. The terminal function does this task by taking a state as input and checking if the state is a terminal state. For the above game, the terminal state is reached when either A or B loses (i.e., either A or B colors a vertex which is connected with another vertex of same color) or all the vertices have been colored (it's a TIE).

For example:
terminal (S_{T1}) = True as vertices 1 and 3 are connected and have Red color (A loses in this case)
terminal (S_{T2}) = True as vertices 2 and 4 are connected and have Blue color (B loses in this case)
terminal (S_X) = False as no two connected vertices have same color.
terminal (S_Y) = False
terminal (S_0) = False

Note that: for the graph shown below with two vertices,
terminal ({(1, Red), (2, Blue)}) = True as all the vertices have been colored and it's a TIE.

Figure 2: Graph B

5. **UTILITY FUNCTION**

This function assigns a utility value (score) for a terminal state. For A to win, the utility value of a terminal state with A winning will be greater than that of the terminal state with B winning. For a TIE, an intermediate utility value will be assigned.

For example:

For graph in Figure 1:

1. utility (S_{T1}) = -10
2. utility (S_{T2}) = +10
3. utility (S_{TIE}) = 0
   (where S_{TIE} is the TIE terminal state if possible (not possible for Graph A in Figure 1))

For graph in Figure 3:

utility ((1, Red), (2, Blue), (3, Red), (4, Uncolored)}) = -1 (A lost)

utility ((1, Blue), (2, Blue), (3, Red), (4, Red)}) = +1 (A won)

utility ((1, Blue), (2, Red), (3, Red), (4, Blue)}) = 0 (TIE)

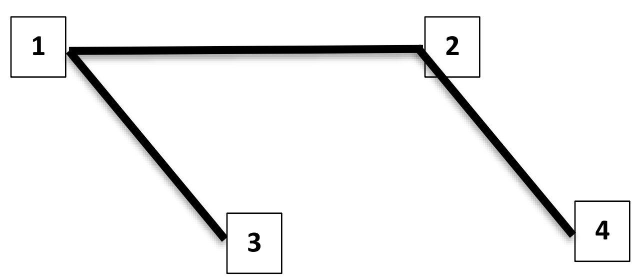

Figure 3: Graph C