Assignment4: Minimax in Mancala game

Aleksandra Ristic
arc19002@student.mdh.se

Utility function

Utility or Evaluation function will give us certain weights to every move. The value from this function represents our leaves in the graph. In figure 2, you can see the utility function used in my solution of problem. There you can notice that my utility function takes into account who is the winner, the number of stones in capture pits and the number of stones on each side, in this order is the priority and importance of these features for the calculation of the evaluation.
Also, it should be mentioned that the possible moves are arranged according to the feature that the move ends on its own in capture pits. In this way are prioritized moves which will give us one more turn, by the rules of the game. In figure 1, you can see how is that done in solution.
Ideas for this definition of utility functions were conceived after analyzing my way of thinking while playing this game.

```
def GetPossibleMoves(tabla, playerTurn):
    moves=[]
    steal=0
    if playerTurn==1:
        for index in range(0,6):
            if tabla[index]!=0:
                #steal=TryCatchOpossit(tabla,playerTurn,index)
                if(steal>0): #not in use
                    tmp=(str(index),steal)
                else:
                    tmp=(str(index),6-index-tabla[index])
                moves.append(tmp)
    else:
        for index in range(7,13):
            if tabla[index]!=0:
                #steal=TryCatchOpossit(tabla,playerTurn,index)
                if(steal>0): #not in use
                    tmp=(str(index),steal)
                else:
                    tmp=(str(index),6-index-tabla[index])
                moves.append(tmp)

    moves.sort(key=lambda tup: tup[1])
    resMoves=[]
    for el in moves:
        resMoves.append(el[0])
    return resMoves
```

Figure 1: Ordering moves

```python
def Evaluation(tabbla,plTn,Iam):  #utility function

    # checking if it is the end of game and who is winner, or they are equal
    #if its not end of game CheckWinner() will return -1
    winner=CheckWinner(tabbla)
    points=0
    #in case that player which is turn is winner of the game, it will get +a lot poin
    # otherwise - points, like sign to dont use that move
    if Iam and winner==plTn:
        points+= 10000
    elif winner==ChangePlayer(plTn):
        points+= -10000
    #if there is not winner, but they have equal winning points
    elif winner==0:
        points+= 5000

    #Calculating number of stones per each side
    stones1=sum(tabbla[0:6])
    stones2=sum(tabbla[7:13])
    points=0
    #calculating difference between captured stones between players and *100
    #calculating difference between no captured stones and *10
    if(plTn==1):
        points+=tabbla[6]*100-tabbla[13]*100
        points+=stones1*10-stones2*10

    else:
        points+=tabbla[13]*100-tabbla[6]*100
        points+=stones2*10-stones1*10

    #In MiniMax function, i am true and i am always looking for max value
    #othervise, if it turn of bot(another player), he is looking for min
    #Because of that if this function is used to check bots evaluation points
    #all points will be *-1
    if not Iam:
        points*=-1
    return points
```

Figure 2: Utility function