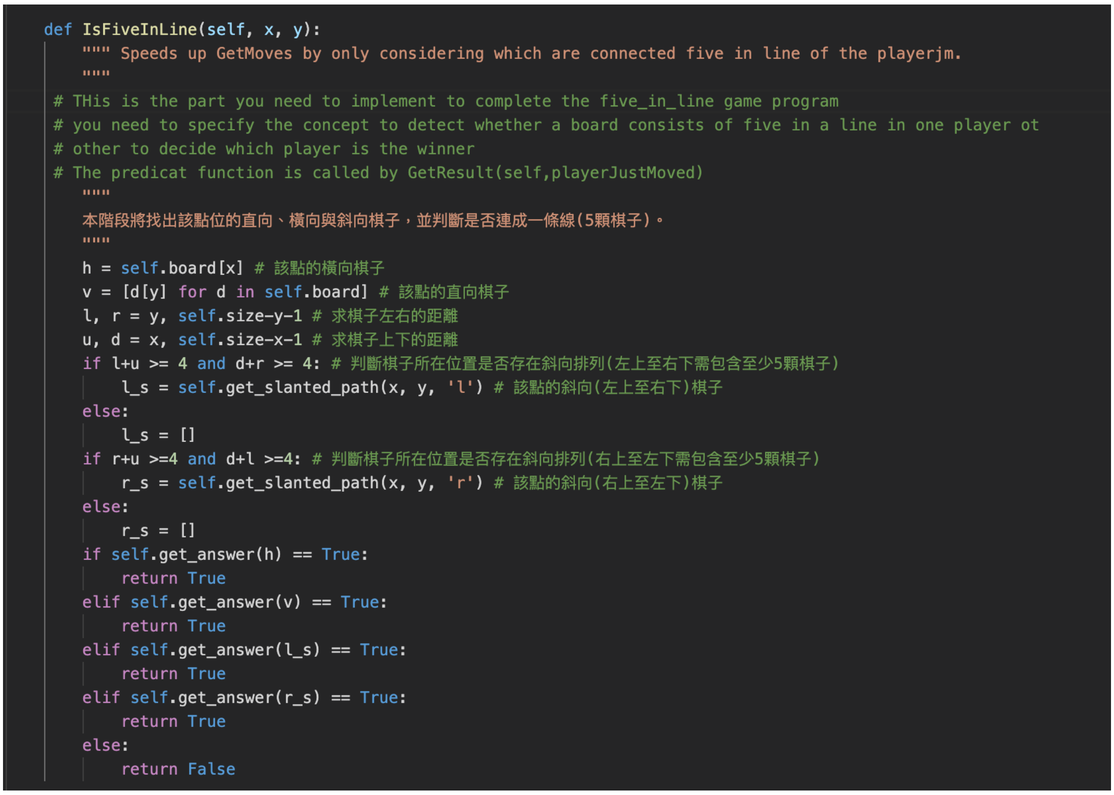
AI project2 assignment2 report

In this project, we are asked to finish IsFiveInLine function by MCTS algorithm. We can use the algorithm try to find the best move.



```python
def IsFiveInLine(self, x, y):
    """ Speeds up GetMoves by only considering which are connected five in line of the playerjm.
    """
 # THis is the part you need to implement to complete the five_in_line game program
 # you need to specify the concept to detect whether a board consists of five in a line in one player ot
 # other to decide which player is the winner
 # The predicat function is called by GetResult(self,playerJustMoved)
    """
    本階段將找出該點位的直向、橫向與斜向棋子，並判斷是否連成一條線(5顆棋子)。
    """
    h = self.board[x] # 該點的橫向棋子
    v = [d[y] for d in self.board] # 該點的直向棋子
    l, r = y, self.size-y-1 # 求棋子左右的距離
    u, d = x, self.size-x-1 # 求棋子上下的距離
    if l+u >= 4 and d+r >= 4: # 判斷棋子所在位置是否存在斜向排列(左上至右下需包含至少5顆棋子)
        l_s = self.get_slanted_path(x, y, 'l') # 該點的斜向(左上至右下)棋子
    else:
        l_s = []
    if r+u >=4 and d+l >=4: # 判斷棋子所在位置是否存在斜向排列(右上至左下需包含至少5顆棋子)
        r_s = self.get_slanted_path(x, y, 'r') # 該點的斜向(右上至左下)棋子
    else:
        r_s = []
    if self.get_answer(h) == True:
        return True
    elif self.get_answer(v) == True:
        return True
    elif self.get_answer(l_s) == True:
        return True
    elif self.get_answer(r_s) == True:
        return True
    else:
        return False
```

Here is my code. We can briefly conclude the MCTS algorithm by the following step!

First: let the current node be the root and expands it to the leaf by simulating a game!

Second: recursively choose the best search

Third: use UCB to calculate the best value

Fourth: repeated until it reached the leaf

Fifth: if leaf never play game before, then we choose randomly to expand a game!

Sixth: based on the outcome, we can update the node and its parent

Seventh: until over the loop limit, recursively do the same thing

Eighth: by doing so, we can get the best move

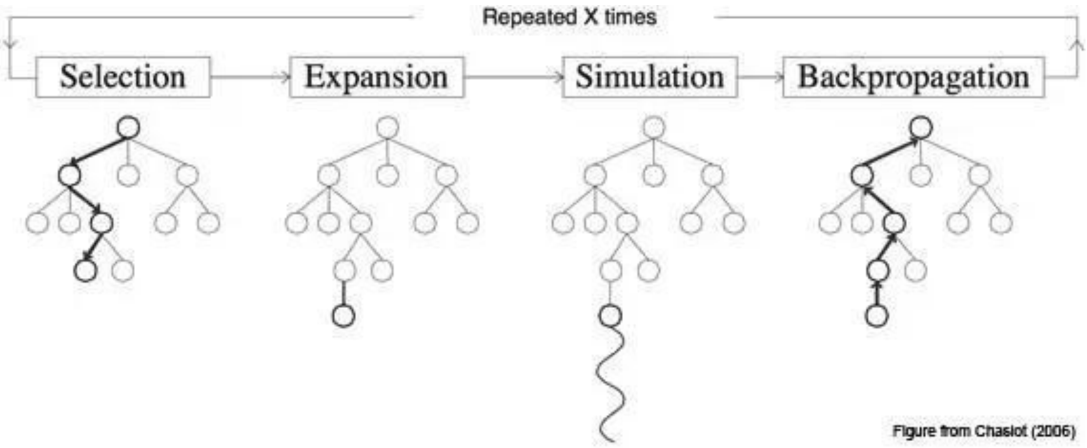


Figure from Chaslot (2006)

In conclusion, this kind of problem consist of four components!

(1) Selection to choose the branch
(2) Expand to next level
(3) Simulation the game
(4) Backpropagation to update the evaluation value