I choosed implementing custom heuristic. My heuristic goes as follows:

```
def score(self, state):
    own_loc = state.locs[self.player_id]
    opp_loc = state.locs[1 - self.player_id]
    own_liberties = state.liberties(own_loc)
    opp_liberties = state.liberties(opp_loc)
    size_of_similar = len([i for i in own_liberties for j in
opp_liberties if i == j])
    return len(own_liberties) - (len(opp_liberties) * 2) +
size_of_similar
```

the heuristic is used to implement blocking the opponent strategy.
Evaluating performance of my agent against baseline:
1.#my_moves - #opponent_moves heuristic

Test Agent	Alpha-Beta Search Algorithm
Greedy	60%
Random	85%
Minimax	40%
Self	50%

2. proposed heuristic

Test Agent	Alpha-Beta Search Algorithm
Greedy	75%
Random	92.5%
Minimax	47.5%
Self	50%

Answers to Questions:
1. my proposed heuristic is completely aggressive and seeks to hunt the opponent and block its movement. Hence, it create an array of those movements that are equal between the player and its opponent and use them in the calculation.
2. clearly, the more depth we visit, the more accurate result we get. But, that is at the expense of time. Adding more time and depth will result in better performance but also, at the expense of efficiency.