Project 3 "Adversarial Search"

Experiment Results

I implemented the minimax algorithm with alpha-beta pruning and iterative deepening. The heuristic score functions tested are:

1. SCORE_1: score = len(my_moves) – len(opponent_moves)
2. SCORE_2: score = len(my_moves) – weight * len(opponent_moves)
3. SCORE_3: for both player I computed a score looking to the liberties of the player itself and if a liberty was in one of the four 2x2 corners I set a +1 to the score, while if it was in the central area of the board a +3 is set, otherwise a +2. The difference between these two scores is taken as final score for the minimax algorithm. The idea behind this scoring function is to privilege the moves that place the knight in the central part of the board because from there the piece has much more liberty to move, while less near the border and even less near the corner.
   score = my_score – weight * opponent_score

Here the results (in term of winning percentage considering fair matches) of the trial games against the sample players (RANDOM, GREEDY, MINIMAX) varying several parameters.

		SCORE_1	SCORE_2	SCORE_3
Depth-limit = 3 Weight = 2 Round = 10	RANDOM	92.5	95	97.5
	GREEDY	65	82.5	82.5
	MINIMAX	50	50	52.5
Depth-limit = 3 Weight = 10 Round = 10	RANDOM		95	95
	GREEDY		80	80
	MINIMAX		47.5	40
Depth-limit = 3 Weight = 2 Round = 100	RANDOM	94	92.2	95.2
	GREEDY	67.8	73.2	79.5
	MINIMAX	50	55	52.8
Depth-limit = 3 Weight = 10 Round = 100	RANDOM		94	
	GREEDY		73.8	
	MINIMAX		48.5	

Fixing now SCORE_3 with weight = 2, we can vary the depth setting also explicitly a timeout to 1 second, and Round = 100:

	RANDOM	GREEDY	MINIMAX
DEPTH = 3	95.2	79.5	52.8
DEPTH = 4	95	83.5	63.5
DEPTH = 5	96	88.5	70.2
DEPTH = 6	96	86.5	72.2

The implemented heuristic SCORE_3 seems to be the best one among the tried ones (and being SCORE_1 the baseline heuristic as proposed during the course). The SCORE_3 heuristic capture the fundamental aspect of the game for which if the knight is at the center of the board it has more possibilities to move in contrast to be on the edges or even worst on the corners. Given that the scoring function is built to incorporate this feature and opposite balancing the two scores of the players with a weight configurable to give more or less importance to the corresponding addendum.

While the first set of experiment is conducted at a fixed depth (3), in the second set of experiments the objective was to investigate how the results vary with respect to the depth. In doing that I set the timeout explicitly to 1 second (and not the default one of 150 milliseconds) just to be sure that the exploration was conducted until the desired depth without stopping to early. How expected, the higher the depth the better the chances to win.