Report : Udacity AIND : Adversarial Search

Rivière Hadrien

Here is a little report on the experiment I ran for the project : adversarial search. More especially I decided to study two different kinds of heuristic after some documentation :

- Heuristic 1 : based on maximizing both available, and remaining moves for the agent, and minimizing the ones available for the opponent.
- Heuristic 2 : based on only maximizing the available and remaining moves for the agent. We are totally ignoring the opponent's move liberties for this heuristic.

Number	Agent	Opponent	Matches	Time limit	Time per game	Win percentage
#1	Heuristic #1	RANDOM	40	150	0.39s	92.5%
#2	Heuristic #2	RANDOM	40	150	0.38s	100%
#3	Heuristic #1	GREEDY	40	150	0.52s	70.0%
#4	Heuristic #2	GREEDY	40	150	0.44s	80.0%
#5	Heuristic #1	MINIMAX	40	150	0.61s	67.5%
#6	Heuristic #2	MINIMAX	40	150	0.56s	55.0%
#7	Heuristic #1	SELF	40	150	0.69s	52.5%
#8	Heuristic #2	SELF	40	150	0.67s	45.0%

As we can see the first Heuristic wins more frequently against MINIMAX heuristic based opponents. However for easier opponents, and on paper, less aggressive opponents (RANDOM for instance), the first heuristic is over-powered by the second one, that tries to maximize the liberties of the agents regardless of the opponent's ones.

Let's also note that from one heuristic to another, the time spend on one game varies not much. All these experiments has been conducted with a depth of 3. I also tried to mess around with this parameter :

Experiment	Agent	Depth	Matches	Time limit	Time per game	Win percentage
#1	Heuristic #1	2	40	150	0.45s	40.0%
#2	Heuristic #1	3	40	150	0.58s	70.0%
#3	Heuristic #1	4	40	150	1.78s	60.0%
#4	Heuristic #1	6	40	500	5.4s	65.0%

For these experiments I used the first heuristic as defined before. As the depth increases, the time necessary for decisions must be increased. For that matter, I had to edit the Time Limit for the final experiment (which we really do not want to increase, this is just to show theory applies in these expriments). Overall, we can see that the agent does not behave well when it tries to anticipate too many episodes. The previous table shows why I chose a depth of 3 for the thorough previous experiment to determine the best heuristic.

Finally, I decided to confront the previous results with results considering the fair_matches tag. This one allows us to conduct experiments without taking into account a lucky start for one player or another. I also decided to use no heuristic for this bunch of experiments to confront the heuristic chosen before with a baseline opponent only taking into account the available moves for the agent and the opponent.

Experiment	Agent	Matches	Opponent	Time limit	Win percentage
#1	Heuristic #1	40	RANDOM	150	96.2%
#2	No heuristic	40	RANDOM	150	90.0%
#3	Heuristic #1	40	GREEDY	150	73.8%
#4	No heuristic	40	GREEDY	150	70.0%
#5	Heuristic #1	40	MINIMAX	150	65.0%
#6	No heuristic	40	MINIMAX	150	38.8%
#7	Heuristic #1	40	SELF	150	50.0%
#8	No heuristic	40	SELF	150	50.0%

As we may see the results seems more consistent with the experiments above. First, the heuristic drawn seems useful since the win percentage are in all cases better than with the baseline heuristic considering my_moves and his_moves.

Plus, we can see that the use of fair-matches tag is useful : the SELF opponent's lines should always display a 50.0% win percentage. This is the case when the fair_matches tag is enabled, but not the case above when it was not. That way we have a better interpretation out of the results. In my opinion, to diagnose how well the heuristic is doing versus the opponent, this tag should always be used.