Report

October 22, 2021

1 Report

1.1 Advanced Search Techniques

I choose to implement Monte Carlo Tree Search(MCTS) and compare it vs. an Iterative deepening with $\alpha\beta$ (ITERAB) pruning algorithm. The MCTS is the basic implementation with light roll-out(i.e. pick a random action) and UCB(Upper Confidence Bound) for node/action selection. The one addition over the basic is that it reuses the tree in consecutive rounds.

I updated the run_match.py so that one can feed both of the custom algorithms like so:

```
python run_match.py -c SELF -o ITERAB -f -r 10 -p 2
python run_match.py -c ITERAB -o MINIMAX -f -r 10 -p 2
```

1.2 Regarding "Fair"

I have used the fair flag for the tests, but I am not sure that it is correct to call it fair. Why? The agents already take turns starting, so unfairness due to first mover advantage is already handled. What the fair flag does result in is really a check on how well agents perform in the initial rounds(and how well it can recover/build from that start). So it simply provides an input regarding whether or not the agent struggles with the first moves or not.

1.2.1 Result

The evaluations where run for 25 rounds and with the -f (fair) setting (i.e. 50 + 50 games). The value in parentheses is from the fair rounds. Random and Greedy (marked with a *) where run with -f -r 10 setting.

Opponent	MCTS	ITERAB
Random*	90% (100%)	80% (90%)
Greedy*	70% (70%)	70% (90%)
MINIMAX	26% (40%)	78% (84%)
MCTS	54% (58%)	82% (84%)
ITERAB	6% (2%)	60% (46%)
MINIMAX(750 ms)	50% (54%)	—
ITERAB(750 ms)	8% (14%)	—

MCTS performs well versus the basic algos but is outperformed by both MINIMAX and ITERAB, the last by quite some distance. I was, honestly, a bit surprised that MCTS underperformed to this degree because its performance vs. the basics was similar. This simply indicates that performance vs. simple players doesn't give a very good measure of how well it will perform when facing very good agents. The performance of MCTS does improve a little if more processing time is allowed, but MINIMAX has a fixed depth of 3 for these tests so that may be an unfair comparison in that case.

My theory for the underperformance of MCTS is that the naive random rollout and the basic UCB for selection(i.e. no heuristics for guidance) requires too many select-expand-simulate-backprop steps to arrive at a good enough decision to beat ITERAB/MINIMAX.