Final Project Report

Saman Kittani and Sparsh Amarnani

December 9rd, 2023

Abstract

A myriad of real-world decision making problems are intractable to solve with exact methods. Heuristic methods (i.e, Methods based on exploration) offer a relatively computationally efficient solution with the caveat of sacrificing optimality. This project applies a variation of Monte Carlo Tree Search (MCTS) to a complex deck building card game, demonstrating the necessity of such methods.

The game has two competing players. Players begin with six cards; five value cards for purchasing, and one special ability card offering unique actions, such as stealing or destroying cards. After selecting an action, players discard their remaining hand and draw three new cards. The large state space, with 168,000 potential starting states and up to 200 possible states from certain actions. This renders exact solutions impractical.

To tackle this vast state space, we implemented a modified MCTS, incorporating progressive widening. With progressive widening, the number of sampled states for a particular state is capped by a dynamic upper limit. If the number of sampled states reaches the upper limit, the algorithm samples from the already sampled possible states. The widening parameter increases as the number of times the action was selected increases. The exact implemented formulation takes the form:

$$L = \theta_1 N(s,a)^{\theta_2} \tag{1}$$

Where L is the limit, $N(s,a)$ is the amount of times a was selected by state s. θ_1 θ_2 are hyperparameters. These parameters control the behavior of the limit. We arbitrarily defined $\theta_1 = 6$ and $\theta_2 = 0.3$. These definitions ensure that, at the worst case, a particular action will only have approx. 60 sampled states. This formulation significantly reduces the breadth of the tree. Consequently, the tree will need less iterations to search deeply into the tree.

To calculate the value of a leaf node, we used "*biased*" random rollouts. The model randomly simulated actions until a terminal state was reached. The simulator was *biased* away from the end turn action. The *bias* was added because deferring a turn would significantly increase rollout calculations, and it's logically a bad action.

Finally, a value of 0.5 was used for the exploration parameter. The exploration parameter, as part of the UCB1 algorithm, was used to balance exploitation and exploration. Our model favored exploitation. This was also decided to mitigate the consequences of the large state space.

While not optimal, the MCTS implementation provides an efficient, real-time solution. It might miss branches that lead to guaranteed victory. Nevertheless, this project demonstrates the effectiveness of MCTS as a powerful tool for solving decision-making problems with vast state spaces.