CS 5180 Reinforcement Learning and Sequential Decision Making - Project Proposal

Roger Fowler

February 25 2025

1 Problem Statement

Wizard is a trick-taking card game for three to six players. Each hand is composed of a number of tricks. After being dealt hands, players must bet exactly how many tricks they will win during the play of the hand. Rules during play may make cards more or less valuable: a randomly assigned trump suit is the most valuable, followed by the suit led by the first player of that trick. Players also must follow the led suit if possible. Whichever player wins the trick leads the next suit. In addition to the normal 52 card deck, there are also 4 Wizard cards which will beat anything except a previous Wizard, and 4 Jester cards which will not beat anything except a following Jester.

The rules of the game are such that a player only receives points if they win exactly their bet number of tricks; otherwise they lose points. The game has no dealer; only the randomness of the deck and the decisions of the players affect the state. For these reasons, the game is complex enough to justify a machine learning effort beyond a toy problem.

2 Outcomes

The desired outcome of the project is to train an agent to play the game Wizard at a capable level. Comparison benchmarks would be useful; a human player for example. An agent which counts cards and plays statistically optimally may also be possible, but 'optimal' play is too complex to be achieved with a purely deterministic agent, so this benchmark is unrealistic. The game provides a score for each hand and for the full game that can be used as a reward.

3 Algorithms and Platforms

Development will be done in the python gymnasium environment, which provides a standard way for the environment and agents to interact. This problem is inherently multi-agent and adversarial; so training will be done by allowing agents to compete against one another.

Due the very high number of game states (60! from the deck order alone) it is infeasible to explore exhaustively, and the chosen algorithm must learn efficiently. Certain situations can be trained specifically - how to bet, for example, or the situation in which the agent may choose the trump suit - but these are specially important choices.

The objective is to use TD learning for this agent. Learning must be efficient due to the complexity of the game interactions. State values may need to be combined though - the combination of cards in hand, what was bet, how many tricks have been won, what the other players bet, how many they have won, etc. demand that the state space be abstracted. A single value in the state might be how many tricks remain to be won out of how many were bet, for example.

Variations of TD should be sufficient to train a basically capable agent in this space. Different versions of state spaces, Q iterations, reward structures, etc. will be tried out, and the final report will compare the effectiveness of these different techniques.

4 Schedule

- March 14th: Functional simulation environment implemented (with human interaction)
- March 21st: Basic agent implemented
- March 28th - April 11th: Agent iteration and fine-tuning
- April 18th: Report written