Creating a Dominion AI using Genetic Algorithms {ethanmok@stanford.edu}

Dominion

Dominion is a deck-building card game. The random nature of the game and the development of different playing strategies makes this game a suitable machine learning application.

Deck Building Games

Deck building games usually involve two key decision points.

1. **Playing** the hand for the current turn (How to best play the cards in hand)

2. **Building** the deck for future turns (How to best build the deck to ensure future turns are good)

Approach

In this project, only a portion of the cards were used to simplify the decision making process.

Playing of cards in hand was done using a fixed strategy.

Building of decks towards **target decks** was performed and optimized using **genetic algorithms**

Features

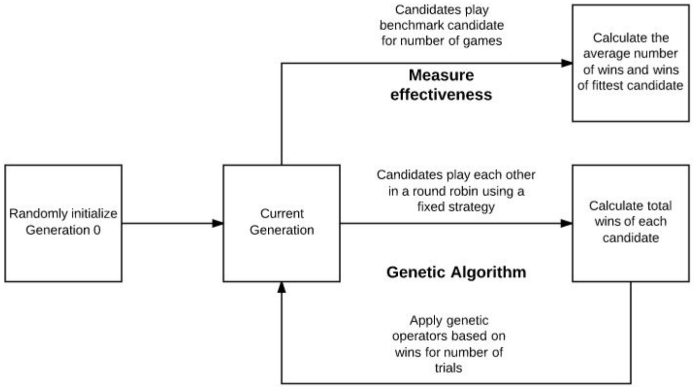
The target deck is customized based on the following features:

For each of the cards,

- Preference of adding the card to deck
- Turn delay before adding of card
- Limit of number of card in deck

This gives 3 features per card, all of which are make up the chromosome. These features are used based on a personal understanding of the game.

Pipeline



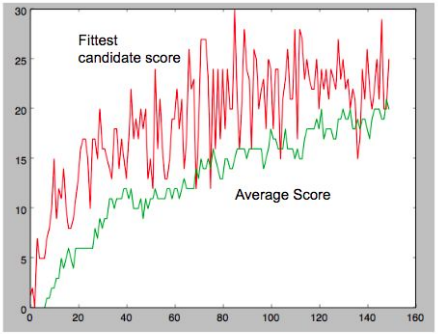
Experiment Setup

150 generations

35 candidates
2 elite candidates
0.2 mutation rate
Uniform crossover
(parents with higher wins have a higher probability of being used)

3 games between candidates in round robin
50 games against previously created benchmark candidate for effectiveness measure

Results



(Out of **50** games against benchmark candidate)

Discussion and Future

The algorithm seems to be working well, with the fittest candidate being able to beat the previous (26 wins) in a 100 generations despite only have 35 candidates per generation.

However, a concern is the large variance in the number of wins against the benchmark candidate. This might be due to the random nature of card games, where luck is a large factor in determining the winner. This could've been solved using a large number of games in future.

Despite the want for additional candidates and games played, the algorithm itself to compute the fitness function (number of wins) is slow because of the requirement for a round robin. This method of computing fitness can be changed in future.

Training a model for the "playing" portion of the game is also encouraged.