# Results of the Blackjacksim Genetic Algorithm

*Abstract*—**The goal of this experiment was to create a genetic algorithm in C++ to find an optimal strategy to win games at the highest percentage. No betting amounts were considered, and results were consolidated into tables of moves that would be done under certain circumstances. The project is likely to be incomplete and requires some better use of randomness to work. However, the main idea is a genetic algorithm can work as signs do show it progressing toward an optimal strategy for the game.**

## I. Introduction

In this experiment a genetic algorithm was used to try and solve the problem of finding the most optimal strategy for Blackjack. Of course, there is no perfect way to play the game as even the best strategies and techniques to play this simple game as the house will win over a long series of rounds. The genetic algorithm used randomly generates several individuals, probability of an individual passing on its move for a given square, mutation percentage for a move square to become an entirely new move, and the percentage of individuals that are used to make new individuals to become part of the next generation. Individuals have a grid of their set moves that you can find an example of in this repository. Individuals use these moves to play a game against the dealer and make certain decisions depending on what cards they possess and what card the dealer is showing face up. Rounds of generations were created by selecting the individuals that won the most games and combining the move sets of these fittest individuals into the move sets for new individuals.

## II. Methods and Experiments

All of the random number generation was done using C++'s Mersenne Twister (seeded by machine's random device) and rand (seeded by time) function calls. These were one of the likely reasons that simulation did not progress toward a good strategy. The experiment involved creating many random individuals and creating a significant number of new generations to see which was the fittest individual in the final generation.

New members of the generations were selected by recombining the move sets of two randomly selected individuals who were the fittest in the generation. These fittest individuals were selected using elitist selection that only let the top percentile create new individuals. After the final generation played their rounds of games the strategy tables were printed out.

The move tables of individuals were randomly created for the first generation. Each individual played a set number of games and then the top individuals created new individuals. These new individuals were created by first mutating a move square with a small chance. If no mutation occurred, one of the two selected parent's move squares would become the move square for the child individual. Individuals all had the moves to hit for a new card, stand with their hand, double down drawing a new card and standing, and splitting their hand into two new hands. After all this simulation the found strategies were produced.

## III. Results

After running simulations, the results found a strategy that had more losses than wins. This was expected because Blackjack strategies are not expected to win for the player and are meant to be a suggestion on what to do when optimal situations to try and win. There were very obvious errors in optimal strategies for certain situations as the fittest individuals suggested extremely bad moves like hitting with a 20 in hand or standing with extremely low scores. However, some of the optimal strategies were beginning to show as clustering of stand moves appeared in rows where the hand is greater than 17. This same trend also showed extremely low scores as well, but the suboptimal moves were still present in final moves.

## IV. Conclusion

The genetic algorithm did show somewhat optimal strategies after 500 generations of simulation where everyone played 200 games. Although, no perfect strategy was found it did look to flow toward an ideal set of moves for playing Blackjack. If more rounds or maybe a more optimal set of parameters to create new generations was input a more optimal strategy could be created. There also is the randomness of the software when creating the initial generation or shuffling the shoe of cards used to play the games. Programming errors may also be present causing the simulations to be imperfect.

Even though the strategy found to play Blackjack was imperfect it followed some expected results with individuals losing more games than they won over a large number of games, and ideal moves of standing with big hands and hitting with small hands. This may suggest that a genetic algorithm can be used to create many individuals and converge toward an optimal set of moves for a series of games.