# Introduction

Ninety-nine is a trick-taking game invented by David Parlett in 1967, the rules to which first appeared in *Games & Puzzles* in 1974 (Parlett, 2017). He revised the rules in 1990 (McLeod, 2003); the revised rules are used in this program. On his website *Original Card Games*, he writes, “Ninety-Nine was designed to meet the need for a skill-demanding Whist-like game for three players, but it works almost as well for four and sports quite satisfactory versions for two and five players” (Parlett, 2017). However, the two-player version is not implemented in this program. Ninety-Nine is not to be confused with the unrelated adding game of the same name (McLeod, 2003).

This author is not aware of any previous research for developing an agent to play ninety-nine.

# Rules

The version of the rules implemented in this program largely reflect Parlett’s preferred way of playing as of the time of this writing. Three players receive 12 cards from a 36-card deck, formed from the standard international deck of 52 cards stripped of all cards lower than the six, while four players receive 13 cards from the full 52-card deck. Five players receive 12 cards from a 60-card superset of the 52-card deck that adds ranks eleven and twelve to all suits; they rank in their expected order between the ten and the jack (Parlett, 2017).

Each player bids in secret, removing three cards from their hand. The suits of the cards discarded in the bidding encode the number of tricks that, when those three cards are summed together, each player must take: each diamond represents zero tricks, each spade represents one trick, each heart represents two tricks, and each club represents three tricks. There is no mechanical meaning to the ranks of the cards discarded in bids; however, there is a strategic meaning to them as bids must be made exactly (Parlett, 2017).

Although declarations and revelations can be made (Parlett, 2017), due to time constraints, the agents are programmed not to do so.

The player to the dealer’s left may lead any card to the first trick. Subsequent players must follow suit if possible; otherwise, they may play any card. If any cards played are trump, then the highest trump card wins; otherwise, the highest card in the suit led wins. In the version of the game programmed here, diamonds are trump for the first hand, with future trump suits (if any) determined by how many players made their bid. Regardless of the outcome, players earn one point per trick taken. In the five-player game, if only one player made their bid exactly, they win 50 additional points; if two players made their bid, they win 40 additional points; if three players made their bid, they win 30 additional points; and so on. Otherwise, if only one player made their bid exactly, they win 30 additional points; if two players made their bid exactly, they win 20 additional points; and if three players made their bid exactly, they win 10 points. If all players in a four-player game make their bid exactly, there is no bonus for doing so. If nobody makes their bid, diamonds are trump for the next hand; if one player makes their bid, the next trump suit is spades; if two players make their bid, the next suit is hearts; if three players make their bid, the next trump suit is clubs; the next hand is played at notrump otherwise. The first to 100 points wins the game, and the first to three games wins the rubber (Parlett, 2017). This program, however, does not use the concept of a rubber.

# Running the program

Upon starting the game, the program prompts for how many players are playing. After being told how many players are playing, it then prompts for a name for each player as well as a strategy for each to take. (The “random” strategy is technically a slight misnomer due to the fact that neither of the completed agents declare or reveal.) After a name and strategy for each player is chosen, the program prompts for how many games to run, up to 10,000. A log of each game is printed in a subfolder called “runs” in the folder that contains the program. Any previous game logs in the folder are overwritten, so if it is desired to save them, they must be copied to another folder before the program is run again.

The program runs correctly when all the agents use the “random” strategy. Although a strategy for implementing Monte Carlo tree search was planned and appears to be able to be chosen, the strategy was not finished before the demonstration, so it is unusable.

The “simple” strategy is an even worse misnomer than the “random” strategy in that it has by far the longest source code of the three planned strategies at 396 lines if blank lines and lines consisting only of comments are ignored. This made it hard to debug, and it was unable to be debugged completely before the demonstration. Among the errors it is known to cause:

* Against all simple agents, it throws a NullPointerException.
* Against at least one random agent, the program throws an ArrayIndexOutOfBoundsException if a simple agent is input as the first player. Otherwise, it is believed to run without throwing an error if the random agents are all input first. In this case, the first hand is believed to always be played correctly, but after a certain point, hands are compromised by the simple agents holding duplicate cards due to cards not being removed properly when playing, as evident in the logs showing cards played more than once under certain circumstances.

# Statistics

The program was run three times to produce samples of 100 three-player games, 100 four-player games, and 100 five-player games with no players making any declarations or revelations, but otherwise playing or bidding randomly.

As the agents play randomly, games take longer in terms of number of hands played than seems reasonable by skilled human ninety-nine players to this author. The average game by three random agents playing against each other took 12.09 hands to finish, slightly longer than the median game length of 12 hands, with a standard deviation of about 3.27 hands. The quickest three-player game lasted 5 hands, while the longest needed 23 to reach a conclusion.

Four random agents playing against each other took about half a hand longer, with an average game being 12.56 hands long with a standard deviation of about 3.46 hands and the median game being 12.5 hands long. The sample maximum with four players was the same, but the sample minimum with four players was 6 hands, albeit achieved six times. In comparison to three- or four-player games, however, five random agents can often finish faster due to a maximum of 59 points being possible without any declarations or revelations as opposed to 39. The average number of hands per game in a sample of 100 five-player games was 9.52 hands, with a median of 9 hands. The standard deviation was higher than in the other two samples, at about 4.58 hands; the five-player sample is home to both the lowest sample minimum of the three samples (three games finished in only two hands, the fewest mathematically necessary to win even the three-player game when declarations and revelations are used) and the highest sample maximum (one game took 27 hands).

# Conclusion

The bugs in the rule-based agent and the lack of completion of the Monte Carlo agent prevent a proper analysis of how they’d perform against the random agent. Fully implemented and working correctly, however, the rule-based agent would probably be expected to perform better than the random agent, while an agent using the Monte Carlo tree search would probably have revealed strategies previously unknown to the author (perhaps some of which would be known by expert players).

# Works Cited

McLeod, J. (2003, November 6). *Ninety-Nine.* Retrieved May 2, 2017, from Card Game Rules: https://www.pagat.com/exact/99.html

Parlett, D. (2017). *Ninety-Nine.* Retrieved May 2, 2017, from Original Card Games: http://www.parlettgames.uk/oricards/ninety9.html