PokerEquity

Brendan Fay (jbf2173) and Lance Wong (ldw2125)

November 2023

1 Background

We intend to make a poker equity calculator, which will rely on a poker hand evaluator. Poker equity is roughly equivalent to the fair value of a particular hand in a particular game; the money in play multiplied by the likelihood of victory. Poker hand evaluation assigns a ranking - essentially a hash - to each hand. These features provide poker players with a fairly complete toolkit for making decisions, and are consequently disallowed from most venues.[1]

2 Implementation

2.1 Poker Hand Evaluator

The poker hand evaluator interprets poker hands as 52-bit binary strings, with each index in the string corresponding to a specific card. If the flushes are handled separately, we can reduce this string to a 13-bit quinary (base 5) number where each bit indicates the number of cards with that rank present in that hand. We can compute a perfect hash of these strings and pattern match to determine their ranking relative to other hands[2][3].

2.2 Poker Equity Calculator

Poker equity is typically computed using the Monte Carlo method. For each hand, a certain number of opponents' hands are randomly generated using the System.Random library. The community cards are then dealt, also randomly, and the winner is determined. This can be determined using the hand evaluator. This experiment is repeated many times - perhaps a few million - and a probability is calculated.

We see several levels of parallelism in this task. First is a purely sequential implementation. Experiments are conducted sequentially, the generation of hands within experiments is conducted sequentially, and the evaluation is done sequentially. We will then add parallel execution for the experiments, then the opponent hand generation, and finally the community card generation.

3 Objective

The primary objective of this project will be to create an efficient, accurate, and parallel poker equity calculator in Haskell. We will benchmark the various versions of our algorithm against each other, and against the fastest C/C++ implementations we have found on the internet.

[1] https://www.pokernews.com/poker-tools/poker-odds-calculator.html

[2] https://github.com/HenryRLee/PokerHandEvaluator/blob/master/Documentation/Algorithm.md

[3] https://joshgoestoflatiron.medium.com/july-17-evaluating-poker-hands-with-lookup-tables-and-perfect-hashing-c21e056da130