Proposal for a
Thesis in the Field of
Information Technology
In Partial Fulfillment of the Requirements
For a Master of Liberal Arts Degree

Harvard University
Extension School
<Submission Date>

Kenneth Ho
[contact information omitted]
kennethho01@fas.harvard.edu

Proposed Start Date: September 21st, 2014
Anticipated Date of Graduation: August 2015
Thesis Director: TBD
1 Tentative Thesis Title
Computing an Approximate Nash Equilibrium for Two-Player Jam/Fold Omaha Hi-Lo Poker Tournaments

2 Abstract
Poker is a popular card game worldwide, with many variations, rules, and structural randomness. It also serves as an excellent structure to investigate problems in multi-agent systems. Recent research in the game of poker have found strategies approaching game theoretically optimal for simplified versions of Texas Hold'em Poker, one of the most popular poker variations. However, there is little research available on another poker variant called Omaha Hi-Lo.

The objective of this project is to solve for an approximate Nash Equilibrium for an abstracted specific poker variant of Omaha Hi-Lo. Specifically, the outcomes of this project will be:

- a computed \( \varepsilon \)-Nash Equilibrium strategy table for the players in the game and a representative set of tournament states, using the Counterfactual Regret Minimization algorithm, and
- an evaluation function for a given hand type and tournament state to approximate the strategy table determined above, similar to the Sit-And-Go Endgame System for Texas Hold'em, for a human player to use the calculated strategy without memorizing multiple tables of 16,432 entries each.

To overcome the computational challenges associated with exploring the much larger game space of Omaha Hi-Lo, this project will use high performance computing tools such as GPGPU with OpenCL, along with cluster computing resources from the Amazon EC2 cloud computing platform.
3 Project Description

3.1 Background

Omaha Hi-Lo is a game in the family of poker games, specifically in the community card variations. In this particular variant, each player is dealt four private cards from a standard deck of fifty-two cards, to be combined with a board of five community cards. Betting rounds occur at four different stages of the hand: immediately after the players are dealt their private four card hands ("preflop"), after dealing the first three community cards onto the board ("flop"), after dealing the fourth community card ("turn"), and after dealing the final community card ("river"). In each betting round, players may bet money into the pot, which others must either match ("call"), increase ("raise"), or forfeit the hand ("fold"). A showdown occurs between each of the remaining players' Hi and Lo hands, using exactly two of the four cards from the player's hand and three cards from the board. The Hi hand is evaluated using standard poker hand rankings, and the Lo hand is evaluated as the lowest set of five cards, ignoring straights, flushes, treating the Ace as the lowest card, with a maximum hand value of an 8-high. The winning Hi hand takes half the pot, and the winning Lo hand takes the other half. A player may win both Hi and Lo to scoop both halves of the pot, constructing their Hi and Lo with different two cards from their hand and three cards from the board.

Poker is a frequent topic of artificial intelligence and game theory research due to its popularity and game structure. In particular, poker has a fundamental structure where players can see their own hands, a piece of information hidden away from opponents. This imperfect information structure lends itself to building strategies that may incorporate deception. An example of deception is "bluffing," holding a weak hand but representing a strong hand through aggressive betting in hopes the opponent will fold. In turn, players can construct strategies that offset the possibility of deception, such as occasionally calling with semi-strong "bluff-catcher" hands that would only beat a bluffing opponent. There exists a balance for using deception and
counter-deception: if a player bluffs with weak hands too often, observant opponents can exploit this by calling and winning with semi-strong hands that can beat weak bluffs. Alternatively, if a player bluffs too little, opponents will adapt by folding to aggressive betting more, causing the player's strong hands to win smaller pots without a showdown. A good strategy must balance the bluffs with strong hands. Researchers in game theory for the game of poker have sought an unexploitable “optimal” strategy. This notion of such a strategy is called a Nash Equilibrium in game theory: a strategy profile, where any player, even knowing everybody’s strategy, cannot increase their individual expected win/loss through exploitation of the other players’ strategies.

Omaha Hi-Lo is structurally similar to Texas Hold'em, the most popular mainstream poker variant. The major similarities are in the use of five community cards and in the betting structure in each round of each hand. However, there are fundamental differences as well, most critically in being dealt four cards instead of two per player, the prescribed use of exactly two of those four cards to construct a player's hands, and the splitting of the pot between a Hi hand and, if any hand qualifies for Lo, a Lo hand.

The additional two cards per player greatly expand the number of possible hands per player. While there are 1,326 possible hands and 169 distinct hand types in Texas Hold'em, there are 270,725 possible hands and 16,432 distinct hand types in Omaha Hold'em and its variants. The much larger number of combinations in Omaha Hold'em creates difficulties in articulating whether a given hand is profitable to play: while pre-flop guidelines are widely published for Texas Hold'em down to each specific hand type (Sklansky & Malmuth, 1999; Harrington & Robertie, 2004), published starting hand guidelines for Omaha are imprecise and descriptive in nature. The Hutchison Point System is one such system for evaluating Omaha hands (Hutchison, 1997). Despite its age and appropriateness only for full-ring cash game conditions, it remains one of the few hand evaluation systems available for the game.
Texas Hold’em, aside from being the most popular poker variant, is also the most heavily researched. A cursory search on Google Scholar returns over 5000 results, including publications by research bodies such as the Association for the Advancement of Artificial Intelligence (AAAI) and the Association for Computing Machinery (ACM). In contrast, there is very little academic research on Omaha Hold’em or any of its variants. A search on Google Scholar search returned no academic articles for Omaha poker, and a search on Harvard’s HOLLIS library system returned zero results for Omaha poker outright.

A major aspect of research into Texas Hold'em is search for a Nash Equilibrium for the game. This research gained ground with the SAGE system (Jones, 2006) that approximates a Nash Equilibrium for a tournament end-game situation. The search also gained a new tool with the development of the Counterfactual Regret (CFR) Minimization algorithm, increasing the feasibly computable search space to $10^{12}$ game states (Zinkevich, Johanson, Bowling, & Piccione, 2007). Researchers have also computed an $\varepsilon$-Nash Equilibrium for two player jam/fold Texas Hold’em (Miltersen & Sørensen, 2007) and three player jam/fold Texas Hold’em (Ganzfried & Sandholm, 2008). The research into Texas Hold'em continues today with the annual Computer Poker Competition and associated workshops at the American Association for Artificial Intelligence conference; the Hyperborean bot, an equilibrium-based Texas Hold’em bot, is still competitive in this venue.

Equilibrium results in the research for Texas Hold'em have used simplifying abstractions to shrink the search space. Notably, the SAGE system, the two player $\varepsilon$-Nash Equilibrium and the three player $\varepsilon$-Nash Equilibrium all made two basic assumptions. The first assumption is that the game structure is that of a tournament, where all players are committed to playing additional hands until they either win all the chips or go broke trying. The second assumption is that the players are constrained to only two possible actions on their turn pre-flop: they either “jam” their
chips all-in for all their chips, or “fold.” The use of these assumptions in building a Texas
Hold'em tournament system is not new, with Sklansky's “The System” (Sklansky, 2002).

Computer speed and problem size remain constraints in building even these simplified
equilibrium strategies. The Texas Hold'em two-player jam/fold equilibrium required two weeks of
CPU time (based on 2007 computer speeds) to compute and verify a solution (Miltersen &
Sørensen, 2007) for a game where each player can hold one of 169 possible hand classes, for a
total of $169 \times 169$ information sets. The three-player Texas Hold'em jam/fold equilibrium required
one month and sixteen processors just to rollout and enumerate the possible three player match-
ups, a total of $169 \times (2 \times 169) \times (3 \times 169)$ information sets, as input data for the equilibrium finding
algorithm (Ganzfried & Sandholm, 2008). A similar analysis for two-player Omaha Hi-Lo
requires computing a strategy over $16,432 \times 16,432$ information sets. This problem would be ten
times larger than the three-player Texas Hold'em problem.

The recent development of OpenCL has the potential of making larger problems solvable
in a reasonable timeframe. OpenCL, released in 2009, is a framework for heterogeneous
computing by Khronos Group. Similar to CUDA by NVIDIA, OpenCL is frequently used to
leverage the parallel processing cores in modern graphics accelerators, performing general
purpose computing on graphics processing units (GPGPU). Speed gains using GPU programming
and OpenCL over CPU-only processes can reach above 100-fold (Stone, Gohara, & Shi, 2010).
The OpenCL platform remains current, with version 2.0 released in November 2013.

A potential speedup of 100 times makes solving abstractions of games as large as Omaha
Hi-Lo possible. The purpose of this project is to determine a score-based approximation to a Nash
Equilibrium for a two-player jam/fold Omaha Hi-Lo Tournament.
## 3.2 Finding a Solution for Omaha Hi-Lo Tournaments

Solving any game requires a rigorous definition of the game, the actions available to the players in the game, the paths that the game can progress based on these actions, and the ending outcomes of those paths. In this project, the Omaha Hi-Lo jam/fold tournament game structure can be described as follows:

1. From their individual chip holdings, Player 1 posts a blind bet of the predetermined “small blind” size, and Player 2 posts a blind bet of the “big blind” size.
2. The game deals four cards from a standard fifty-two card deck to each player. The two players may look at their own hands.
3. Player 1 decides either to “jam” or “fold.” If Player 1 folds, then Player 2 wins from Player 1 chips the amount of the small blind, and this round of the game ends. Otherwise, Player 1 jams, raising to either all his chips or to an amount equal to Player 2’s total chip holding (including big blind).
4. If Player 1 jams in Step 3, then Player 2 decides either to “call” or “fold.” If Player 2 folds, then Player 1 wins from Player 2 chips the amount of the big blind, and the round ends. Otherwise, Player 2 calls the raise by Player 1.
5. If Player 2 calls, the game deals five community cards. It then determines the winner of the Hi hand and the Lo hand (if applicable), and determines the round’s result as one of Player 1 wins, Player 1 quarters, Draw, Player 2 quarters, or Player 2 wins.
6. If either Player 1 or Player 2 wins, they take their opponent’s contribution to the pot. If Player 1 or Player 2 quarters, they take half of their opponent’s contribution to the pot. Otherwise, neither player wins anything.
7. If either player now has all the chips, the tournament is over and that player is the winner. Otherwise, another round begins, with Player 1 and Player 2 switching positions (Player 2 acts first, and posts the small blind, etc.).
The construction of a solver for the game defined above depends on constructing high performance components to handle the mechanical and analytical aspects of the game. Specifically, a very fast Hand Evaluator and an efficient Equilibrium Solver are necessary to solve this Omaha Hi-Lo jam/fold tournament.

### 3.2.1 Hand Evaluator

The Hand Evaluator is necessary to determine the probability of each result should both players decide to play their hands. That information affects the decisions by both players. For example, if Player 1 receives a weak hand that is likely to lose against Player 2 if both players jam, Player 1 may prefer to fold and take the certain but small loss, rather than risk incurring a large loss. Fully defining the chance nodes in the game’s extensive form requires determining all the probability distributions for each of those $16,432 \times 16,432$ chance nodes.

The open-source poker-eval library contains an Omaha Hi-Lo hand evaluator. However, it is not fast enough. Using the library to evaluate one match-up requires three seconds, which would make evaluating all the nodes take twenty-five CPU-years. To evaluate all the possible match-up classes in one month using a cluster of five computers, an evaluator must achieve
twenty match-ups per second, equivalent to a hand evaluation speed of 21,720,160 Omaha hands per second.

This project proposes aggressively pre-computing hand results and simplifying hand classes to speed up the evaluation process. Consider an Omaha five card poker hand. Of the \( \binom{52}{5} = 2598960 \) possible five-card poker hands, there are 7,462 possible Hi poker hand values and 56 possible Lo poker hand values, a combination that we can encode in a thirty-two bit integer value. We can break the hand down further into the two player’s cards and three community cards. There are \( \binom{52}{2} = 1326 \) possible combinations of two cards. However, if the first two cards are not suited, we can immediately establish that the final hand will not be a flush or straight flush, and therefore the suits of the two cards become irrelevant in determining the value of the hand. This reduces the number of hand classes to \( 13 + \binom{13}{2} \times 5 = 403 \), for the pairs and for the non-pairs associated with suit information: \{unsuited, spades, hearts, diamonds, clubs\}.

Similarly, we can pre-evaluate the \( \binom{52}{5} = 2598960 \) possible combinations of community cards into different classes, keeping suit information for flush potential:

- \( \binom{13}{5} \times 4 = 5148 \) flushes
- \( \binom{13}{4} \times 4 \times 13 = 37180 \) four-flushes and a card in a different suit of any value
- \( \binom{13}{3} \times 4 \times \binom{14}{2} = 104104 \) three-flushes and two cards of any value
- \( \binom{17}{5} - 13 = 6175 \) hands with which no flush can be constructed

The above reduces the number of relevant hand classes of five cards down to 152,607. This is a small enough number that a table of size \( 152,607 \times 403 \times 4 = 246,002,484 \) bytes (under 256 megabytes) can be constructed. With this table and other pre-computed tables, we can reduce an Omaha Hi-Lo game round into:
• Six index calculations of two-card holdings for the four cards in a player’s hand, plus

• Six index calculations for the other player’s hand, plus

• For each of $\binom{44}{5} = 1086008$ possible five-card rollouts:
  
  o One index calculation for the community cards, plus

  o One table lookup $\{1..2598960\} \rightarrow \{1..152607\}$ to convert a community card combination to a community card class, plus

  o Twelve table lookups $\{1..152607, 1..403\} \rightarrow \{1..7462, 0..56\}$ to determine the possible Hi and Lo hands given the players’ hands and community cards

This totals 14,118,104 random access table lookups and 1,086,020 table index calculations per match-up. Initial testing of this configuration suggests this number of lookups and calculations can be done in less than 0.5 seconds, for a speed of two match-ups per second.

Adding OpenCL can provide even more speed. Aside from delivering multi-threaded processing “for free” with the built-in support for multi-core processing, it also provides the ability to hide memory latency. Latency hiding occurs when a GPU performs a memory lookup, and while the lookup is processing, for the GPU to continue processing on a separate independent thread. This has the effect of dramatically lowering the cost of random memory access, since processing continues during that access time. Initial testing of the same algorithm as before, except using GPU multi-threading and latency hiding, suggests a match-up execution time of 0.012 seconds, corresponding to an evaluation speed of more than eighty match-ups per second.

I therefore expect that constructing a hand evaluator that can process the desired twenty match-ups per second is feasible.
3.2.2 Equilibrium Solver

The problem of finding a Nash Equilibrium for a general game lies in the complexity class of PPAD-complete. Scholars consider this class of problems to require more than polynomial time to solve. However, the task of finding an approximation to a Nash Equilibrium is simpler in execution, using the concept of regret and regret minimization.

Regret, loosely defined in this context, is the score that a player foregone (whether by losing instead of winning, losing more than their minimum loss, or winning less than their maximum gain) by following their strategy versus following a best response to the opponent’s strategy. An update mechanism can then evolve the players’ strategies in a self-play function, incorporating these best responses in future play iterations to lose less, gain more, and reduce regret. This method of iteratively using better responses, if it converges to a strategy over time, converges to a Nash Equilibrium (Nisan, Roughgarden, Tardos, & Vazirani, 2007).

The iterative improvements to a strategy converge to a Nash Equilibrium, with regret decreasing proportional to the number of actions available and the square root of the number of iterations. Ganzfried and Sandholm found that fewer than 100 rounds of fictitious play are necessary to bound regret to less than 0.01 for a particular tournament round (Ganzfried & Sandholm, 2008), and that a full tournament (presuming the value of a round given the stack sizes and positions of the players) can be computed in “several hours” (Ganzfried & Sandholm, 2008).

My initial benchmark for this calculation completes a computation of the best response against an opponent’s strategy in less than 0.1 seconds. I expect that determining an $\varepsilon$-Nash Equilibrium for this restricted Omaha Hi-Lo tournament is feasible.
3.3 Articulating the Solution

The equilibrium solution for an Omaha Hi-Lo jam/fold tournament would be a prescribed (though potentially randomized, for mixed strategies) action for a player, given the game situation. The game situation, in turn, consists of the player’s hand classification, chip holdings per player, and the player’s position of Small Blind or Big Blind. Miltersen and Sørensen, in their analysis of two-player jam/fold Texas Hold’em, published results for 160 different chip holdings scenarios and one blinds level (Miltersen & Sørensen, 2007); their strategy function \( \{1.169, 1.160, 1.2\} \rightarrow \{[0,1],[0,1]\} \) has a domain size of 54,080 possibilities. Even this is already difficult to articulate, and it is impractical to use in live situations; players do not have the luxury of referring to strategy printouts in a tournament. A similar output for an Omaha Hi-Lo tournament would require a table of 5,258,240 different situations, a table that would be virtually impossible to memorize.

The original SAGE system, in contrast, provides a scoring rule that takes those same game conditions, adds an additional dimension (ratio of big blind to player stack size), and reduces it to seven calculations and two table lookups. Though it is technically inferior compared to the equilibrium solution determined by Miltersen and Sørensen, it is simpler to use, handles more situations, and generates strong results.

This project proposes constructing a similar scoring rule for Omaha Hi-Lo tournaments, to score hands and recommend whether to jam or fold. This requires building a binary classifier (jam or fold) for approximately 30 million samples in dimensions potentially including:

- Player Position
- Size of Big Blind
- Total number of chips among all players
- Number of chips held by each player
- **Hand Attributes, such as:**
  - Suited-ness (flush potential, for Hi hands)
  - Suited-ness High Card (flush potential, flush power)
  - Suited-ness Spread (straight flush potential)
  - Paired-ness (high pair, set, full house, quads potential)
  - Paired-ness High Card (high pair, set power)
  - Rundown (straight potential)
  - Rundown Gap count (straight potential)
  - Card Values (general hand power)
  - Low Card Count (Lo hand potential)
  - Low Card Values (Lo hand power)

Two methods of generating a binary classifier are to use Linear Regression and to use Support Vector Machines (SVM). Linear regression is a standard method for estimating the coefficients for a linear function over a number of variables. In the case of evaluating an Omaha Hi-Lo hand, we can attempt to determine coefficients for variables corresponding to the hand and game attributes, construct the linear scoring function, and then use it to determine whether to jam or fold, based on whether the function returned a positive or negative number.

The other method listed, support vector machines, is more capable of handling non-linear inputs. This method is theoretically grounded, popular, and effective. With pre-analysis of the data, this method can also be fast, with non-linear SVM classification time potentially as low as two minutes for 16 million samples (Fehr, Arreola, & Burkhardt, 2008). Similar to the linear regression method, the target result is a simple function that recommends whether to jam or fold, based on whether the function classifies the situation as TRUE or FALSE.
With well-established binary classifier algorithms for large data sets comparable to that currently proposed, I conclude that constructing a binary classifier and scoring system for Omaha Hi-Lo hands is feasible.

### 3.4 Resource Constraints

Computing resources will be the most likely and severe constraint for this project. This project proposes using a cluster of high-performance servers to perform the intensive computation of match-up rollouts and equilibrium solving. Still, these constraints are manageable, as the Amazon EC2 platform makes such computers available inexpensively.

Amazon EC2 provides GPU servers on a per-hour basis at a reasonable price. Subject to maximum server limits, Amazon charges a normal price of $0.65 USD per server per hour for a “g2.2xlarge” server equipped with a compute-optimized NVIDIA GPU, and $2.10 USD per server hour for a “cg1.4xlarge” server for twice the performance. However, these servers are frequently available at much lower spot prices, subject to market demand. The spot price for instances of the “g2.2xlarge” servers hovers around $0.065 per hour, a discount of 90%, and $0.135 per hour for “cg1.4xlarge” servers. Amazon provides an API to retrieve historical spot pricing, and third party sites such as ec2price.com publish that data for easy reference. The estimated cost of the GPU cluster on EC2 for three months is $1000.

### 4 Work Plan

There are five milestones proposed in this project:

1. Developing a fast Omaha Hi-Lo hand evaluator, capable of evaluating at least twenty match-ups per second.

2. Pre-computing the results of each of the possible $16432 \times 16432$ matchups as a cache for an Equilibrium Solver,
3. Developing an Equilibrium Solver that harnesses the capabilities of modern GPUs and OpenCL,

4. Computing an $\epsilon$-Nash Equilibrium for the tournament using the cached data and the Equilibrium Solver, and

5. Clustering hands and constructing a hand scoring heuristic for Omaha Hi-Lo, similar to the SAGE System for Texas Hold'em, using a classifier system such as a support vector machine, and the equilibrium determined above.

### 4.1 Assumptions, Risks and Alternatives

The major assumption in this project lies in the feasibility of the computation itself. The major risk factor in this project is the resource constraints dictated by the thesis project. These constraints include the thesis time limit of nine months and the availability and cost of using Amazon EC2.

The final aspect of the work plan also assumes that the 16,432 classes of hands are well ordered for constructing a score-based hand ranking heuristic, and that determining this order is tractable within the time and resource constraints.

### 4.2 Preliminary Schedule

<table>
<thead>
<tr>
<th>Date</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>September 21\textsuperscript{st}, 2014</td>
<td>Thesis proposal approved. Planning begins on new hand evaluator for Omaha Hi-Lo (Phase 1).</td>
</tr>
<tr>
<td>October 7\textsuperscript{th}</td>
<td>Development begins on new hand evaluator.</td>
</tr>
<tr>
<td>October 27\textsuperscript{th}</td>
<td>Development complete for new hand evaluator. Planning begins on</td>
</tr>
<tr>
<td>Date</td>
<td>Activity</td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>November 21&lt;sup&gt;st&lt;/sup&gt;</td>
<td>Development begins on hand versus hand matchup system.</td>
</tr>
<tr>
<td>December 15&lt;sup&gt;th&lt;/sup&gt;</td>
<td>Execution of hand versus hand match-ups on Amazon EC2 begins. Planning begins on building the Equilibrium Solver (Phase 3).</td>
</tr>
<tr>
<td>February 7&lt;sup&gt;th&lt;/sup&gt;</td>
<td>Equilibrium Solver development complete. Equilibrium Solver execution begins. Planning begins on the hand classifier (Phase 5).</td>
</tr>
<tr>
<td>March 1&lt;sup&gt;st&lt;/sup&gt;</td>
<td>Development begins on the hand classifier.</td>
</tr>
<tr>
<td>March 10&lt;sup&gt;th&lt;/sup&gt;</td>
<td>Equilibrium Solver execution (Phase 4) complete.</td>
</tr>
<tr>
<td>March 25&lt;sup&gt;th&lt;/sup&gt;</td>
<td>Development of the hand classifier complete. Execution begins on the hand classifier.</td>
</tr>
<tr>
<td>April 1&lt;sup&gt;st&lt;/sup&gt;</td>
<td>Execution of the hand classifier complete. Thesis paper write-up begins.</td>
</tr>
<tr>
<td>May 1&lt;sup&gt;st&lt;/sup&gt;</td>
<td>Thesis paper draft submitted to advisor.</td>
</tr>
<tr>
<td>May 21&lt;sup&gt;st&lt;/sup&gt;</td>
<td>Thesis revised and final draft submitted.</td>
</tr>
</tbody>
</table>
5 Glossary

CUDA: Compute Unified Device Architecture, a platform model developed by NVIDIA Corporation to facilitate general purpose computing on their graphics accelerators.

Flush: a poker hand consisting of five cards all in the same suit.

Four-flush: a poker hand with exactly four cards in the same suit, as an attribute to evaluate flush potential with additional cards.

GPGPU: General Purpose computing for Graphics Processing Units.

Heterogeneous computing: computing using more than one kind of processor, potentially with different types of processors having different specializations or efficiency in particular computing tasks.

Information set: a set of possible game states, as observed by a particular player, based on the common knowledge of all other players' strategies and the previously observed game history.

Nash Equilibrium: a set of strategies for each player in a game, such that with each player knowing their own strategy and the strategies of each of the other players, no player can improve their expected outcome by unilaterally changing their strategy.

$\varepsilon$-Nash Equilibrium: a set of strategies for each player in a game, such that with each player knowing their own strategy and the strategies of each of the other players, no player can improve their expected outcome by more than a bounded small score $\varepsilon$ by unilaterally changing their strategy.

OpenCL: Open Compute Language, a standard programming interface for parallel computing across dissimilar computing hardware, including normal CPUs, GPUs, and accelerator boards.
Rundown: an Omaha hand consisting of four cards of consecutive value, or near consecutive value.

SAGE System: an end-game strategy for a Texas Hold'em tournament, developed by Lee Jones and James Kittock, that scores a player's hand and recommends whether to raise all-in or fold, based on the tournament conditions and hand score.

Straight: a poker hand consisting of five cards of consecutive value.

Straight Flush: a poker hand consisting of five cards of the same suit of consecutive value, meeting the requirement for the hand to be a flush and to be a straight.

Three-flush: a poker hand with exactly three cards in the same suit, as an attribute to evaluate flush potential with additional cards.

6 References

6.1 Works Cited


6.2 Works Consulted


