

The Mathematics Behind Euchre

An Honors Thesis (HONRS 499)

by

Jonathon J. Wright

Thesis Advisor

John Emert

A handwritten signature in black ink that reads "John Emert". The signature is written in a cursive style with a large, looping initial "J".

Ball State University

Muncie, Indiana

May 4, 2001

Expected Date of Graduation

May 2001

Sp 2011
Thesis
LD
2459
.24
2001
.W75

Acknowledgements

The author would like to thank several people who have assisted in the completion of this honors thesis. Special thanks go out to Professor John Emert for his support and guidance throughout the completion of this project and also his proof reading efforts. Gratitude also goes to Dr. Kerry Jones for his direction concerning the Monte Carlo system used to model the playing out of the hands of Euchre. Without your help and assistance this thesis would never have become a reality.

Abstract

This thesis delves into the complex card game of euchre in an effort to display the mathematics behind the game. There are two major elements of euchre where the mathematics will be explored. The first is the shuffling of the 24 cards in the euchre deck. This will entail using normal distribution along with the random number generator and matrix multiplication. The final aspect will be to model how a given hand will be played out assuming the players understand how to play. This will be done using a Monte Carlo system, as there are too many possible ways to play to feasibly model them all. Therefore, the Monte Carlo system will allow the modeling using the random function and one hundred thousand iterations, without contemplating theoretically every possible combination of how the hands could be played.

Table of Contents

I.	The History of Euchre.....	1.
II.	The Rules of Euchre.....	2.
III.	Modeling the Shuffling of Euchre.....	5.
IV.	Monte Carlo System.....	8.
V.	Monte Carlo System Graphs.....	13.
VI.	Conclusions.....	16.
	Appendix.....	17.
	References.....	32.

The History of Euchre

Numerous card historians have determined that euchre is a direct descendant of the Spanish game Triomphe. Nevertheless, euchre also shares similarities to another European card game of German origin called Juckerspiel. An early version of euchre was played extensively in France during the 1700s under the name Ruff.

During the reign of Napoleon in Europe, euchre was modernized and brought to America in the French controlled New Orleans. From Louisiana the game traveled up the path of the Mississippi River into the northern states where it gained considerable popularity. Nearly 100 years ago, euchre was the number one card game in the United States. Even so, a few decades later, the game's popularity succumbed to bridge, spades, and hearts. However, euchre is still adored by many. There is a national organization and there are several web sites devoted solely to the game. Even though euchre is not the most popular card game in the United States, it is still tremendously admired in the Midwest states of Indiana, Ohio, Illinois, Wisconsin, and Michigan.

The Rules of Euchre

The rules of euchre include several variations. The most common way to play utilizes four players as two teams of two and a pack of 24 cards created by using only nines through Aces of a regular deck of 52. Another variation consists of using a pack of 32 cards produced by using sevens through Aces of a regular deck of 52. There are many other variations to the game such as three-handed euchre, two-handed euchre, railroad euchre, and call ace euchre. However, I will focus my attention on the most popular version.

The highest trump is the jack, called the right bower. The second highest is the other jack of the same color, called the left bower. For instance, if spades are trump, the jack of spades is the right bower and the jack of clubs is the left bower. The rest of trump suit ranks A (third-best), K, Q, 10, 9. For non-trump suits, the rank is A (high), K, Q, J (if not left bower), 10, and 9. Each player is dealt five cards in-groups of 3-2 or 2-3, and the dealer must adhere to their original order of dealing. The succeeding card in the pack is flipped face up on the table and is referred to as the turn-up card.

The turn-up card proposes the trump suit for that deal, but only becomes trump if one of the players accepts it. Starting with the player on the dealer's left, each player in turn may pass or accept the suit of the turn-up card. An opponent of the dealer accepts by saying, "Pick it up." The dealer accepts by saying, "I pick it up." If all four players pass, the turn-up card is turned face down. Then

each player in turn has a second chance to pass or name a trump suit. The suit named must be different than the suit of the turn-up card. This process does not have to make it all the way around the table to each player; it stops as soon as someone makes trump.

Whoever chooses the trump suit by either accepting the turn-up card or declaring the suit in the second round, becomes the caller. The caller has the right to say, "I play alone," becoming a lone caller, in which case his partner discards his hand and stays out of play.

If the turn-up card is accepted, the dealer has the right to use it as part of his hand, exchanging it for a card in his hand. The player to the left of the dealer makes the opening lead. If there is a lone caller, the player to the lone caller's left has the first lead. The hands are played out in five tricks. A player must follow suit if able; if unable to follow suit, he may play another card. A trick is won by the highest trump played to it, or by the highest card of the suit led if no trump is present. The winner of the trick leads the next trick.

Only the team winning three or more tricks scores. Winning all five tricks is called a march. When the calling team fails to win the majority, they have been euchred. The calling side scores one point for three or four tricks or two points for a march. A caller playing alone scores one point for three or four tricks or four points for a march. Opponents of the caller score two points for a euchre. Usually, each team keeps track of their points won by using two low cards, such as a six and a four, by exposing a dot for each point won. The team to reach a total of ten points first wins the game.

The same dealer re-deals if a card is exposed or if the pack is found to be incorrect. If a player leads out of turn and the trick is gathered, it stands as regular. Otherwise the bad lead becomes an exposed card and other cards played to it may be retracted without penalty. At the next proper turn of the offending side to lead, the opponent to his right may name the suit to be led. This penalty does not apply to a lone player, but he may be required to retract a lead out of turn. If a player (not playing alone) exposes a card from his hand except in proper play, he must leave it face up on the table and must play it at his first legal chance.

If a player looks at a quitted trick or gives illegal information to his partner, then the next time the offending team has the lead, the opponent to the right of the leader may name the suit to be lead. Furthermore, failure to follow suit when able is a renege. A player may correct his renege before a trick is gathered. Otherwise it stands as an established renege, meaning the opponents of the offender may score two points.

Modeling the Shuffling of Euchre

I have set out to model the shuffling of the most common version of euchre where there are 24 cards using visual C++. The first step in this process is to create a random deck of cards. To accomplish such a task I set up a one-dimensional array (vector) with 24 positions containing values incremented by one all the way to 24. Each one of these 24 numbers represents one of the 24 cards used in the game. Whereas one is the ace of hearts, two is the king of hearts, and so on for all suits. Once this vector was set up, I utilized a for-loop from one to 24 which randomly chose the first card and moved it to the first position of another array and continued randomly choosing cards without repeat until all 24 cards were randomly located in the new array. From here I had a random deck of cards to begin the game with, but I needed to shuffle them.

The next step in modeling card shuffling is to determine where the deck of cards will be split. Usually the deck is approximately split in half and the cards are then shuffled. To model this, normal distribution appeared to work very well using a mean of 12.5 and a variance of 3. Subsequently, this causes the majority of card splitting to occur between cards 10 through 15 as a split any worse would most likely be noticed by the shuffler.

Once the deck is split, the shuffler has a stack of cards in their right and left hand. When shuffling, I assumed that the cards weren't evenly shuffled. I assumed that at most, two cards from the left or the right stack were piled

together. I then used the random number generator to decide whether one or two cards from either the left or right stack would be used. Then I alternated between the right and left deck until the cards were all piled upon one another.

To actually model this, I commenced with a 24 by 24 identity matrix. When this matrix is multiplied by the randomly chosen deck, the result is the randomly chosen deck. The reason for this is that the one in each row of the identity matrix when multiplied by the randomly chosen deck, determines the resulting position of the randomly chosen deck. The identity matrix does not change the original position. Therefore, I needed to manipulate the arrangement of the identity matrix rows to produce a 24 by 24 shuffle matrix.

The split row of the identity matrix was moved to the bottom of the shuffle matrix occasionally with the preceding identity row. Then the bottom row of the identity matrix followed to the bottom (above previously defined rows) of the shuffle matrix also occasionally with the preceding identity matrix row. This continued until one stack ran out first and the remaining rows of the identity matrix were placed in the top rows of the shuffle matrix. To shuffle the cards, the random deck array was multiplied by the 24 by 24 shuffle resulting in a newly shuffled deck.

The following is a diagram of the shuffling process:

Initialized Deck of Cards

[1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24]

Randomly Chosen Deck

[9 2 18 5 6 22 8 10 3 11 24 13 1 14 15 23 16 17 19 12 20 7 21 4]

24 by 24 Identity Matrix

```
[1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1]
```

24 by 24 Shuffle Matrix

```
[0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0]
[1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0]
[0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1]
```

*	Randomly Chosen Deck	=	Shuffled Deck
	[9]		[5]
	[2]		[6]
	[18]		[8]
	[5]		[10]
	[6]		[24]
	[22]		[13]
	[8]		[14]
	[10]		[15]
	[3]		[17]
	[11]		[20]
	[24]		[21]
	[13]		[4]
	[1]		[9]
	[14]		[2]
	[15]		[18]
	[23]		[22]
	[16]		[63]
	[17]		[11]
	[19]		[1]
	[12]		[23]
	[20]		[16]
	[7]		[19]
	[21]		[12]
	[4]		[7]

Monte Carlo System

Quite often in the field of mathematics it is extremely difficult to exactly model situations. For instance, when trying to find the area of an irregular region, it is tricky to find the actual equation to model the border. An individual could spend a tremendous amount of time attempting to find such an equation. However, if the irregular region were circumscribed in a square of known area where one hundred random shots were fired, the proportion of shots hitting the region would give a good approximation of the area. In addition, the more shots fired the more exact the area approximation would be.

However, this method wouldn't be very useful for modeling the area of a square. The exact area of the square could easily be found by measuring the length of one side and squaring it. The method of random shots isn't exact and would take much longer to model the area.

The example of using many random shots to model situations that are extremely complex is the basis of Monte Carlo Systems. These systems utilize random events to model situations and these events are run numerous times. This is a reliable method as many random events develop of good picture of what is actually occurring.

The final endeavor of this honors thesis entails determining how a given deal of euchre will be played out. Due to the multitude of possible ways the deal could be played, a Monte Carlo System was used. Such systems use the random

number generator, coded strategy, and numerous iterations to model what the probability of either team will be to win the hand with their given cards.

To set up this Monte Carlo System I utilized visual basic. The initial step included setting up each of the four players with five cards. Player one and three are on a team versus players two and four. For the system, player one received the nine and Ace of Clubs, the nine and Ace of Spades, and the nine of hearts. Player two was given the ten and jack of clubs, nine and ten of diamonds, and the ten of spades. Player three was furnished with the ten and jack of hearts, jack of spades, jack of diamonds, and the queen of clubs. In addition, player four was bequeathed the queen of hearts, queen and king of spades, queen of diamonds, and king of clubs. However, the Monte Carlo System will function properly for any valid euchre deal.

Next, the Dealer needed to be determined. The dealer for every hand is chosen randomly as they could be any of the four players and many iterations will allow on average all four players to deal equally. With the dealer selected, the leader is subsequently chosen, as they are the person to the left of the dealer. Therefore if player four dealt, then player one would be the leader.

The leader has many options available at their disposal. First of all they may either lead trump or not. This was set to be random as long as they had trump in their hands. The reasoning behind allowing this to be random is that they are both equally likely strategies depending on the person's playing style and should be modeled so as to reflect this. If the leader plays trump then they will play their highest trump in hopes of winning the trick while removing the other

players of their trump. Playing their lowest trump wasn't modeled, as this would usually cause their trump to be beaten by another resulting in the loss of one of their highest valued cards.

If the leader decides to not play trump, then they have the choice of either playing their highest non-trump card or their lowest. They would play their highest in hopes that others will have to follow suit allowing them to win the trick. By playing the lowest card they would in effect be relying on their partner to win the trick and most likely be throwing away their card. This choice to either play a high or low non-trump card was also allowed to be random as both options are exercised frequently.

This brings us to the first follower. This person must follow suit if possible. They have the option of either playing their lowest follow suit card if they can't beat the leader or their highest if it is better than the leader. If they can't follow suit then they have the option of playing trump as long as the lead card isn't trump. They will want to play their lowest trump card so as to maximize the value of their highest if they do have better. In addition, they wouldn't play trump if they had the right bower, as this is an automatic trick and usually used to take out high trump cards such as the left bower. If their hand doesn't contain trump, then the follower would want to play the lowest card in their hand.

The subsequent follower plays by the same principles as the first follower, but they must also take into account what the follower and leader have already played. For instance, if they have to follow suit and nothing in their hand beats what has been played then they will play their lowest follow-suit card. In

addition, they will not trump in if the follower has already done so and follower two can't beat follower one with their trump. If so, they will play their lowest card. Follower three plays just like the other followers except that they must taken into account what the leader, follower one, and follower two have played.

This is the backbone of the logic behind the leader and follower relationship. The random number generator is used to model the leader as their decisions can vary greatly. However, the follower really doesn't have much random choice as to what they will play. These followers play by strategy.

To better model all the possible outcomes, this scenario for the given hand was run for all four values of trump. The number of tricks won by each team for all four values of trump using the given cards was recorded. Then such a model was ran for a multitude of iterations.

After 10,000 iterations, player one and three which I will call team one, won 79 percent of all the hands whereas player one and two called team two, took only 21 percent of the hands. Looking a little deeper, when hearts were trump, team one won every single hand. This makes sense as player three had the right and left bower along with the ten of hearts while his partner had the nine of hearts. In addition team two combined with only the queen of hearts. In addition, when diamonds were trump team one dominated with 97 percent as player three had both bowers.

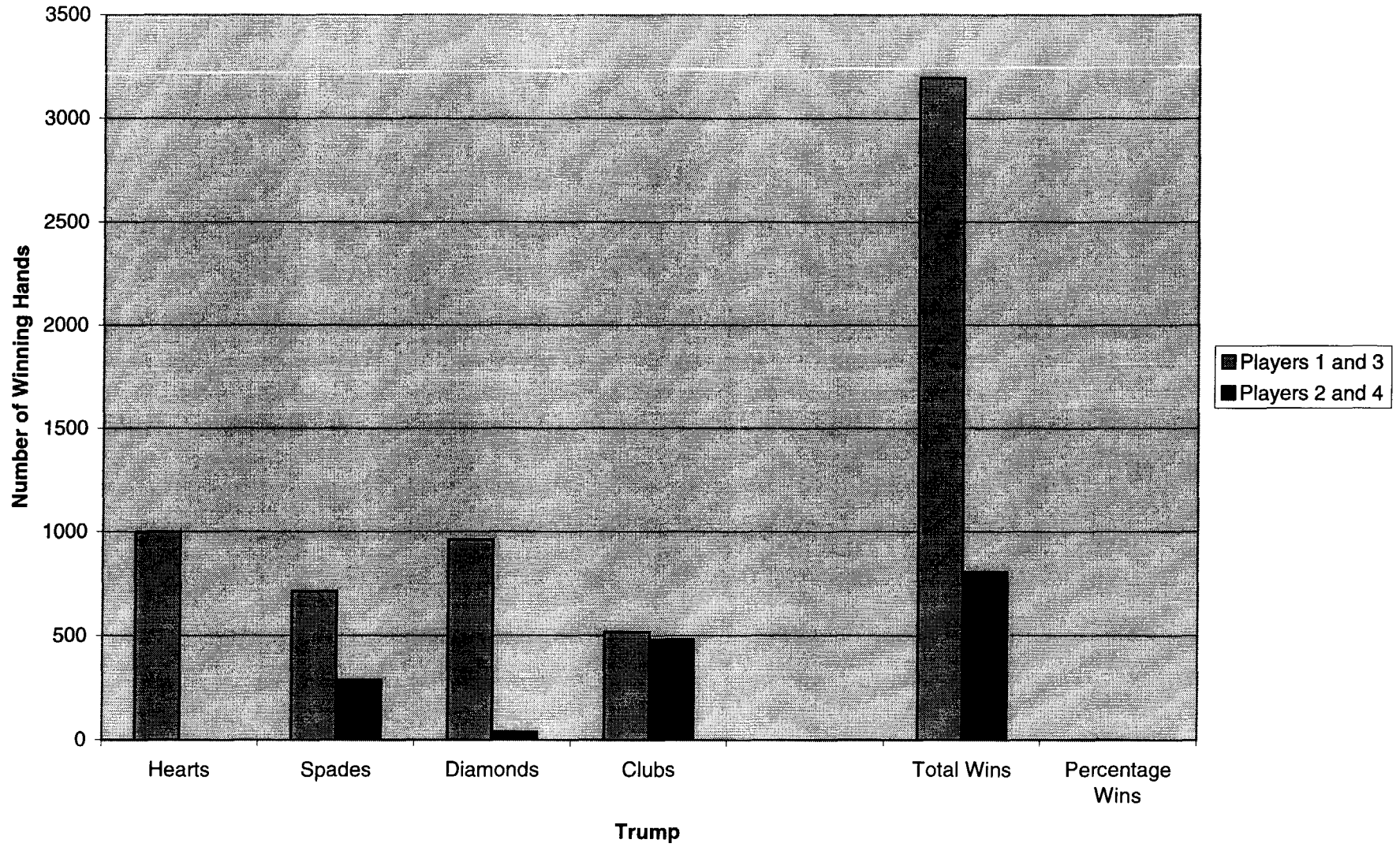
When spades were trump, team one won 70 percent while team two picked up only 30. This appears to be correct as team one had the 9 and Ace of spades, and the right bower. While team two had the 10, queen, and king of

spades. Team one had overall better cards, but depending on how they were played, team two still had a chance.

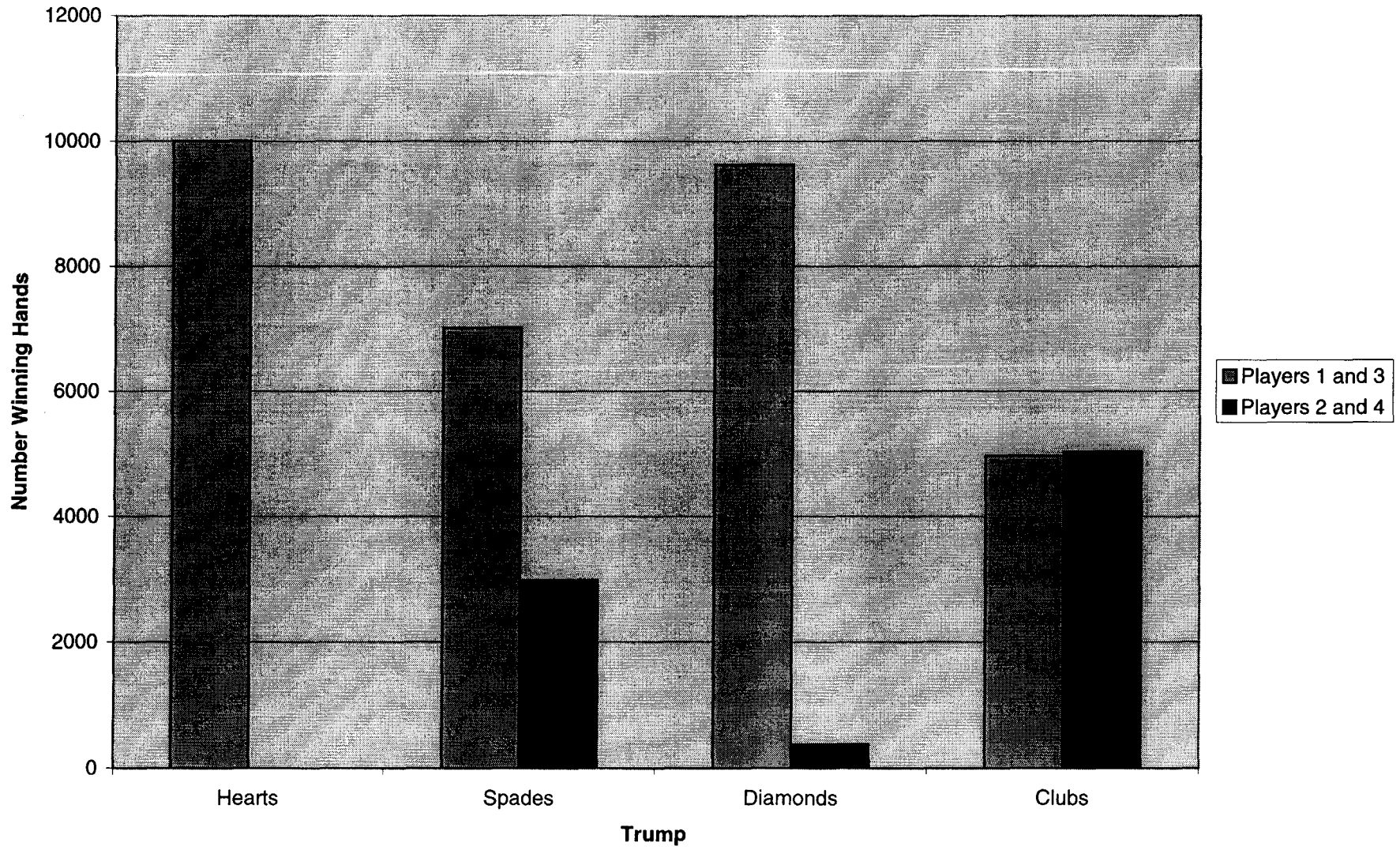
Nevertheless, with clubs trump the wins were almost evenly split with team one taking a little over 50 percent of the hands. The reason behind this is that both teams had nearly even trump hands. Team one possessed the 9, queen, and ace of clubs, and the left bower. Team two combined for the 10, and king of clubs, and the right bower.

Iterations of 100, 1000, 10000, and 100000 were run to compare the results. Samples with a smaller number of iterations had a greater variance. This is due to the law of large numbers. The more times you run the system the less the variance and the closer the answer is to the exact value. Nonetheless, all four sets of iterations were extremely close with the maximum difference of a little more than one percent. Therefore, a huge amount of iterations isn't necessary, as the system is very random and not exceptionally biased.

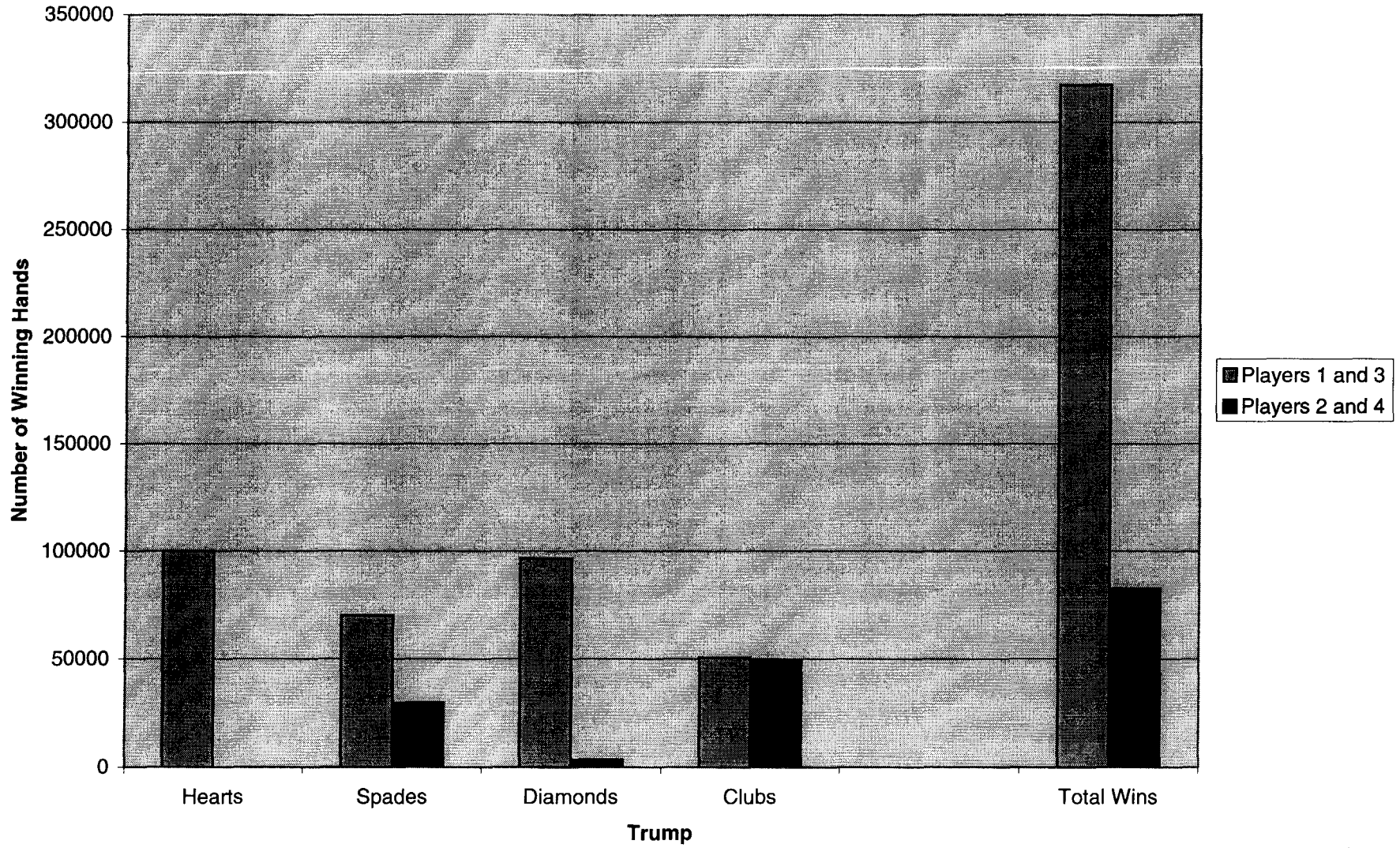
Number of Winning Hands 1000 Iterations



Number of Winner Hands With 10000 Iterations



Number of Winning Hands 100,000 Iterations



Conclusions

At first glance, the underlying mathematics behind the game of euchre didn't seem extremely complicated. My first endeavor of modeling card shuffling was a little complicated, but overall it was rather easy. I assumed modeling how a set of cards would be played out would have a similar difficulty level. However, this was not the case. I began writing the Monte Carlo System in visual basic and expected it to be just a few pages. To my surprise the code ended up being more than ten pages. The system worked as I expected, but I never imagined the logic behind such a Monte Carlo System would be so long and complicated.

However, one major surprise with the Monte Carlo System was the accuracy of very few iterations. The percentage of wins using 100 and 100,000 iterations were almost identical. The reasoning behind this is due to the fact that the system is very random and not significantly biased, therefore eliminating the need for a large number of iterations.

Module1 - 1

'Appendix A

Sub MonteCarlo()

'(Player Number, Cards Num, Value = (1-6), Suit = (H=1, S=2, D=3, C=4)

Dim Cards(4, 6, 2), TurnCard(1, 1, 2), HandWinner(100000, 4), TrickWinner(5, 4) 'five tricks and
four possible trumps

Dim NumHeartTrumpWin(2), NumSpadeTrumpWin(2), NumDiamondTrumpWin(2), NumClubTrumpWin(2)

Iteration2 = 100000

For Iteration = 1 To Iteration2

Randomize

For Trump = 1 To 4

'Set up Players Hands

'Player 1

Cards(1, 1, 1) = 1

Cards(1, 1, 2) = 4 '4

Cards(1, 2, 1) = 1

Cards(1, 2, 2) = 2 '2

Cards(1, 3, 1) = 6

Cards(1, 3, 2) = 2 '2

Cards(1, 4, 1) = 6

Cards(1, 4, 2) = 4 '4

Cards(1, 5, 1) = 1

Cards(1, 5, 2) = 1

'Player 2

Cards(2, 1, 1) = 2

Cards(2, 1, 2) = 4 '4

Cards(2, 2, 1) = 2

Cards(2, 2, 2) = 2 '2

Cards(2, 3, 1) = 3 '3

Cards(2, 3, 2) = 4 '4

Cards(2, 4, 1) = 1 '1

Cards(2, 4, 2) = 3 '3

Cards(2, 5, 1) = 2

Cards(2, 5, 2) = 3

'Player 3

Cards(3, 1, 1) = 3

Cards(3, 1, 2) = 2

Cards(3, 2, 1) = 4

Cards(3, 2, 2) = 4

Cards(3, 3, 1) = 2

Cards(3, 3, 2) = 1

Cards(3, 4, 1) = 3

Cards(3, 4, 2) = 1

Cards(3, 5, 1) = 3

Cards(3, 5, 2) = 3

'Player 4

Cards(4, 1, 1) = 4

Cards(4, 1, 2) = 2 '2

Module1 - 2

Cards(4, 2, 1) = 5
Cards(4, 2, 2) = 2 '2

Cards(4, 3, 1) = 5 '5
Cards(4, 3, 2) = 4 '4

Cards(4, 4, 1) = 4 '4
Cards(4, 4, 2) = 3 '3

Cards(4, 5, 1) = 4
Cards(4, 5, 2) = 1

'Changes Values of Trump Bowers from 3 to 8 and 7 for right and left respectively
For i = 1 To 4

For j = 1 To 5

Select Case Trump
Case 1

If Cards(i, j, 2) = 1 And Cards(i, j, 1) = 3 Then
Cards(i, j, 1) = 8
End If

If Cards(i, j, 2) = 3 And Cards(i, j, 1) = 3 Then
Cards(i, j, 1) = 7
Cards(i, j, 2) = 1
End If

Case 2

If Cards(i, j, 2) = 2 And Cards(i, j, 1) = 3 Then
Cards(i, j, 1) = 8
End If

If Cards(i, j, 2) = 4 And Cards(i, j, 1) = 3 Then
Cards(i, j, 1) = 7
Cards(i, j, 2) = 2
End If

Case 3

If Cards(i, j, 2) = 3 And Cards(i, j, 1) = 3 Then
Cards(i, j, 1) = 8
End If

If Cards(i, j, 2) = 1 And Cards(i, j, 1) = 3 Then
Cards(i, j, 1) = 7
Cards(i, j, 2) = 3
End If

Case 4

If Cards(i, j, 2) = 4 And Cards(i, j, 1) = 3 Then
Cards(i, j, 1) = 8
End If

If Cards(i, j, 2) = 2 And Cards(i, j, 1) = 3 Then
Cards(i, j, 1) = 7
Cards(i, j, 2) = 4
End If

End Select

Next j

Next i

Module1 - 3

Select Case Trump

Case 1

Suit = "H"

Case 2

Suit = "S"

Case 3

Suit = "D"

Case 4

Suit = "C"

End Select

'Randomly Chooses Dealer and subsequently chooses Leader

Dealer = Int(4 * Rnd) + 1

If Dealer = 4 Then

Leader = 1

Else

Leader = Dealer + 1

End If

For Trick = 5 To 1 Step -1

'Leader plays trump or not

'See if Leader has trump

NumTrump = 0

For i = 1 To Trick

If Cards(Leader, i, 2) = Trump Then

NumTrump = NumTrump + 1

End If

Next i

'if TrumpOrNot = 1 lead trump if have or have all trump, else don't lead trump

TrumpOrNot = Int(2 * Rnd) + 1

If (NumTrump > 0) And (TrumpOrNot = 1) Or NumTrump = Trick Then

'Plays Leaders Highest Trump

TempLeaderValue = 0

TempLeaderSuit = 0

If NumTrump > 0 Then

For i = 1 To Trick

If Cards(Leader, i, 2) = Trump Then

If Cards(Leader, i, 1) > TempLeaderValue Then

TempLeaderValue = Cards(Leader, i, 1)

TempLeaderSuit = Cards(Leader, i, 2)

End If

End If

Next i

End If

Else

'Leader doesn't play trump

Module1 - 4

```
For i = 1 To Trick
    If Cards(Leader, i, 2) <> Trump Then
        TempLeaderValue = Cards(Leader, i, 1)
        TempLeaderSuit = Cards(Leader, i, 2)
        Exit For
    End If
Next i

HighOrLow = Int(2 * Rnd) + 1

If HighOrLow = 1 Then
    'Play lowest card

    For i = 1 To Trick

        If Cards(Leader, i, 1) < TempLeaderValue And Cards(Leader, i, 2) <> Trump Then
            TempLeaderValue = Cards(Leader, i, 1)
            TempLeaderSuit = Cards(Leader, i, 2)
        End If

    Next i

Else
    'Play highest card
    For i = 1 To Trick

        If Cards(Leader, i, 1) > TempLeaderValue And Cards(Leader, i, 2) <> Trump Then
            TempLeaderValue = Cards(Leader, i, 1)
            TempLeaderSuit = Cards(Leader, i, 2)
        End If

    Next i

End If

End If

'Followers

If Leader = 4 Then
    Follower = 1
Else
    Follower = Leader + 1
End If

NumFollowSuit = 0

'Find the number of cards the follower has in hand to follow suit
For i = 1 To Trick

    If Cards(Follower, i, 2) = TempLeaderSuit Then
        NumFollowSuit = NumFollowSuit + 1
    End If

Next i

If NumFollowSuit > 0 Then

    'Find card in deck that can follow suit
    For i = 1 To Trick

        If Cards(Follower, i, 2) = TempLeaderSuit Then
            TempFollowerValue = Cards(Follower, i, 1)
            TempFollowerSuit = Cards(Follower, i, 2)
        End If

    Next i

End If
```

Module1 - 5

```
Next i

'Find the largest and smallest Follow suit card
TempFollowerBigValue = TempFollowerValue
TempFollowerSmallValue = TempFollowerValue

For j = 1 To Trick

  If Cards(Follower, j, 2) = TempLeaderSuit Then
    If Cards(Follower, j, 1) >= TempFollowerBigValue Then
      TempFollowerBigValue = Cards(Follower, j, 1)
      TempFollowerBigSuit = Cards(Follower, j, 2)
    End If

    If Cards(Follower, j, 1) <= TempFollowerSmallValue Then
      TempFollowerSmallValue = Cards(Follower, j, 1)
      TempFollowerSmallSuit = Cards(Follower, j, 2)
    End If

  End If

Next j

'Play the largest if beats leader, else play smallest
If TempFollowerBigValue > TempLeaderValue Then
  TempFollowerPlayValue = TempFollowerBigValue
  TempFollowerPlaySuit = TempFollowerBigSuit
Else
  TempFollowerPlayValue = TempFollowerSmallValue
  TempFollowerPlaySuit = TempFollowerSmallSuit
End If

Else
'Can't follow suit as don't have any
'Usually want to play trump as long as it isn't the right bower, play smallest
NumTrumpF = 0

For i = 1 To Trick

  If Cards(Follower, i, 2) = Trump Then
    NumTrumpF = NumTrumpF + 1
  End If

Next i

If NumTrumpF > 0 Then

  For i = 1 To Trick

    If Cards(Follower, i, 2) = Trump Then
      TempFollowerValue = Cards(Follower, i, 1)
      TempFollowerSuit = Cards(Follower, i, 2)
    End If

  Next i

  'Find the largest and smallest trump cards
  TempFollowerBigValue = TempFollowerValue
  TempFollowerSmallValue = TempFollowerValue

  For j = 1 To Trick

    If Cards(Follower, j, 2) = Trump Then
      If Cards(Follower, j, 1) >= TempFollowerBigValue Then
        TempFollowerBigValue = Cards(Follower, j, 1)
        TempFollowerBigSuit = Cards(Follower, j, 2)
      End If

      If Cards(Follower, j, 1) <= TempFollowerSmallValue Then
```

Module1 - 6

```
        TempFollowerSmallValue = Cards(Follower, j, 1)
        TempFollowerSmallSuit = Cards(Follower, j, 2)
    End If

    End If

Next j
'Play the smallest trump as long as it isn't the right bower
If TempFollowerSmallValue <> 8 Then
    TempFollowerPlayValue = TempFollowerSmallValue
    TempFollowerPlaySuit = TempFollowerSmallSuit
End If
Else
'Want to play lowest card that suit doesn't match others in hand
TempFollowerPlayValue = Cards(Follower, 1, 1)
TempFollowerPlaySuit = Cards(Follower, 1, 2)

For i = 1 To Trick

    If Cards(Follower, i, 1) <= TempFollowerPlayValue Then
        TempFollowerPlayValue = Cards(Follower, i, 1)
        TempFollowerPlaySuit = Cards(Follower, i, 2)
    End If

Next i

End If

End If

'Follower2

If Follower = 4 Then
    Follower2 = 1
Else
    Follower2 = Follower + 1
End If

NumFollow2Suit = 0

'Find the number of cards follower2 has in hand to follow suit
For i = 1 To Trick

    If Cards(Follower2, i, 2) = TempLeaderSuit Then
        NumFollow2Suit = NumFollow2Suit + 1
    End If

Next i

If NumFollow2Suit > 0 Then

'Find card in deck that can follow suit
For i = 1 To Trick

    If Cards(Follower2, i, 2) = TempLeaderSuit Then
        TempFollower2Value = Cards(Follower2, i, 1)
        TempFollower2Suit = Cards(Follower2, i, 2)
    End If

Next i

'Find the largest and smallest Follow suit card
TempFollower2BigValue = TempFollower2Value
TempFollower2SmallValue = TempFollower2Value

For j = 1 To Trick

    If Cards(Follower2, j, 2) = TempLeaderSuit Then
        If Cards(Follower2, j, 1) >= TempFollower2BigValue Then
```

Module1 - 7

```
        TempFollower2BigValue = Cards(Follower2, j, 1)
        TempFollower2BigSuit = Cards(Follower2, j, 2)
    End If

    If Cards(Follower2, j, 1) <= TempFollower2SmallValue Then
        TempFollower2SmallValue = Cards(Follower2, j, 1)
        TempFollower2SmallSuit = Cards(Follower2, j, 2)
    End If

End If

Next j

'Play the largest if beats leader and Follower, else play smallest
If TempFollower2BigValue > TempLeaderValue And TempFollower2BigValue > TempFollowerPlay
value Then
    TempFollower2PlayValue = TempFollower2BigValue
    TempFollower2PlaySuit = TempFollower2BigSuit
Else
    TempFollower2PlayValue = TempFollower2SmallValue
    TempFollower2PlaySuit = TempFollower2SmallSuit
End If

Else
'Can't follow suit as don't have any
'Usually want to play trump as long as it isn't the right bower and can beat other playe
d trump, play smallest
'Find number of trump in Follower2's hand
NumTrumpF2 = 0

For i = 1 To Trick

    If Cards(Follower2, i, 2) = Trump Then
        NumTrumpF2 = NumTrumpF2 + 1
    End If

Next i

If NumTrumpF2 > 0 Then

    For i = 1 To Trick

        If Cards(Follower2, i, 2) = Trump Then
            TempFollower2Value = Cards(Follower2, i, 1)
            TempFollower2Suit = Cards(Follower2, i, 2)
        End If

    Next i

'Find the largest and smallest trump cards
TempFollower2BigValue = TempFollower2Value
TempFollower2SmallValue = TempFollower2Value

For j = 1 To Trick

    If Cards(Follower2, j, 2) = Trump Then
        If Cards(Follower2, j, 1) >= TempFollower2BigValue Then
            TempFollower2BigValue = Cards(Follower2, j, 1)
            TempFollower2BigSuit = Cards(Follower2, j, 2)
        End If

        If Cards(Follower2, j, 1) <= TempFollower2SmallValue Then
            TempFollower2SmallValue = Cards(Follower2, j, 1)
            TempFollower2SmallSuit = Cards(Follower2, j, 2)
        End If

    End If

Next j
```

```
'Play the smallest trump as long as it isn't the right bower and can beat Follower
If TempFollower2SmallValue <> 8 Then
```

```
  If TempFollower2SmallSuit <> TempFollowerPlaySuit Then
    TempFollower2PlayValue = TempFollower2SmallValue
    TempFollower2PlaySuit = TempFollower2SmallSuit
```

```
  Else
```

```
    If TempFollower2SmallValue > TempFollowerPlayValue Then
      TempFollower2PlayValue = TempFollower2SmallValue
      TempFollower2PlaySuit = TempFollower2SmallSuit
```

```
    Else
```

```
      If TempFollower2BigValue > TempFollowerPlayValue Then
        TempFollower2PlayValue = TempFollower2BigValue
        TempFollower2PlaySuit = TempFollower2BigSuit
```

```
      Else
```

```
        TempFollower2PlayValue = Cards(Follower2, 1, 1)
        TempFollower2PlaySuit = Cards(Follower2, 1, 2)
```

```
        For i = 1 To Trick
```

```
          If Cards(Follower2, i, 1) <= TempFollower2PlayValue Then
            TempFollower2PlayValue = Cards(Follower2, i, 1)
            TempFollower2PlaySuit = Cards(Follower2, i, 2)
          End If
```

```
        Next i
```

```
      End If
```

```
    End If
```

```
  End If
```

```
Else
```

```
'Play smallest card if only have right bower as trump
TempFollower2PlayValue = Cards(Follower2, 1, 1)
TempFollower2PlaySuit = Cards(Follower2, 1, 2)
```

```
For i = 1 To Trick
```

```
  If Cards(Follower2, i, 1) <= TempFollower2PlayValue Then
    TempFollower2PlayValue = Cards(Follower2, i, 1)
    TempFollower2PlaySuit = Cards(Follower2, i, 2)
  End If
```

```
Next i
```

```
End If
```

```
Else
```

```
'Want to play lowest card thats suit doesn't match others in hand
TempFollower2PlayValue = Cards(Follower2, 1, 1)
TempFollower2PlaySuit = Cards(Follower2, 1, 2)
```

```
For i = 1 To Trick
```

```
  If Cards(Follower2, i, 1) <= TempFollower2PlayValue Then
    TempFollower2PlayValue = Cards(Follower2, i, 1)
    TempFollower2PlaySuit = Cards(Follower2, i, 2)
  End If
```

```
Next i
```

```
End If
```

```

End If

'Follower3

If Follower2 = 4 Then
    Follower3 = 1
Else
    Follower3 = Follower2 + 1
End If

NumFollow3Suit = 0

'Find the number of cards follower3 has in hand to follow suit
For i = 1 To Trick

    If Cards(Follower3, i, 2) = TempLeaderSuit Then
        NumFollow3Suit = NumFollow3Suit + 1
    End If

Next i

If NumFollow3Suit > 0 Then

    'Find card in deck that can follow suit
    For i = 1 To Trick

        If Cards(Follower3, i, 2) = TempLeaderSuit Then
            TempFollower3Value = Cards(Follower3, i, 1)
            TempFollower3Suit = Cards(Follower3, i, 2)
        End If

    Next i

    'Find the largest and smallest Follow suit card
    TempFollower3BigValue = TempFollower3Value
    TempFollower3SmallValue = TempFollower3Value

    For j = 1 To Trick

        If Cards(Follower3, j, 2) = TempLeaderSuit Then
            If Cards(Follower3, j, 1) >= TempFollower3BigValue Then
                TempFollower3BigValue = Cards(Follower3, j, 1)
                TempFollower3BigSuit = Cards(Follower3, j, 2)
            End If

            If Cards(Follower3, j, 1) <= TempFollower3SmallValue Then
                TempFollower3SmallValue = Cards(Follower3, j, 1)
                TempFollower3SmallSuit = Cards(Follower3, j, 2)
            End If

        End If

    Next j

    'Play the largest if beats leader and Follower and Follower2, else play smallest
    If TempFollower3BigValue > TempLeaderValue And TempFollower3BigValue > TempFollowerPlay
value And TempFollower3BigValue > TempFollower2PlayValue Then
        TempFollower3PlayValue = TempFollower3BigValue
        TempFollower3PlaySuit = TempFollower3BigSuit
    Else
        TempFollower3PlayValue = TempFollower3SmallValue
        TempFollower3PlaySuit = TempFollower3SmallSuit
    End If

Else

    'Can't follow suit as don't have any
    'Usually want to play trump as long as it isn't the right bower, play smallest and will
beat all other cards

```

Module1 - 10

```
NumTrumpF3 = 0
```

```
For i = 1 To Trick
```

```
    If Cards(Follower3, i, 2) = Trump Then  
        NumTrumpF3 = NumTrumpF3 + 1  
    End If
```

```
Next i
```

```
If NumTrumpF3 > 0 Then
```

```
    For i = 1 To Trick
```

```
        If Cards(Follower3, i, 2) = Trump Then  
            TempFollower3Value = Cards(Follower3, i, 1)  
            TempFollower3Suit = Cards(Follower3, i, 2)  
        End If
```

```
    Next i
```

```
'Find the largest and smallest trump cards  
TempFollower3BigValue = TempFollower3Value  
TempFollower3SmallValue = TempFollower3Value
```

```
For j = 1 To Trick
```

```
    If Cards(Follower3, j, 2) = Trump Then  
        If Cards(Follower3, j, 1) >= TempFollower3BigValue Then  
            TempFollower3BigValue = Cards(Follower3, j, 1)  
            TempFollower3BigSuit = Cards(Follower3, j, 2)  
        End If
```

```
        If Cards(Follower3, j, 1) <= TempFollower3SmallValue Then  
            TempFollower3SmallValue = Cards(Follower3, j, 1)  
            TempFollower3SmallSuit = Cards(Follower3, j, 2)  
        End If
```

```
    End If
```

```
Next j
```

```
'Play the smallest trump as long as it isn't the right bower and can beat all other tr
```

ump

```
If TempFollower3SmallValue <> 8 Then  
    If TempFollower3SmallSuit <> TempFollowerPlaySuit Then  
        TempFollower3PlayValue = TempFollower3SmallValue  
        TempFollower3PlaySuit = TempFollower3SmallSuit  
    Else
```

```
        If TempFollower3SmallValue > TempFollowerPlayValue Then  
            TempFollower3PlayValue = TempFollower3SmallValue  
            TempFollower3PlaySuit = TempFollower3SmallSuit  
        Else
```

```
            If TempFollower3BigValue > TempFollowerPlayValue Then  
                TempFollower3PlayValue = TempFollower3BigValue  
                TempFollower3PlaySuit = TempFollower3BigSuit  
            Else  
                TempFollower3PlayValue = Cards(Follower3, 1, 1)  
                TempFollower3PlaySuit = Cards(Follower3, 1, 2)
```

```
For i = 1 To Trick
```

```
    If Cards(Follower3, i, 1) <= TempFollower3PlayValue Then  
        TempFollower3PlayValue = Cards(Follower3, i, 1)  
        TempFollower3PlaySuit = Cards(Follower3, i, 2)  
    End If
```

Module1 - 11

```
        Next i
      End If
    End If
  End If

Else
  'Play smallest card if only have right bower as trump
  TempFollower3PlayValue = Cards(Follower3, 1, 1)
  TempFollower3PlaySuit = Cards(Follower3, 1, 2)

  For i = 1 To Trick

    If Cards(Follower3, i, 1) <= TempFollower3PlayValue Then
      TempFollower3PlayValue = Cards(Follower3, i, 1)
      TempFollower3PlaySuit = Cards(Follower3, i, 2)
    End If

    Next i

  End If

Else
  'Want to play lowest card thats suit doesn't match others in hand
  TempFollower3PlayValue = Cards(Follower3, 1, 1)
  TempFollower3PlaySuit = Cards(Follower3, 1, 2)

  For i = 1 To Trick

    If Cards(Follower3, i, 1) <= TempFollower3PlayValue Then
      TempFollower3PlayValue = Cards(Follower3, i, 1)
      TempFollower3PlaySuit = Cards(Follower3, i, 2)
    End If

    Next i

  End If

End If

'Removes Played cards
For i = 1 To Trick

  If Cards(Leader, i, 1) = TempLeaderValue And Cards(Leader, i, 2) = TempLeaderSuit Then

    For j = i To Trick
      Cards(Leader, j, 1) = Cards(Leader, j + 1, 1)
      Cards(Leader, j, 2) = Cards(Leader, j + 1, 2)
    Next j

    Cards(Leader, Trick, 1) = ""
    Cards(Leader, Trick, 2) = ""

  End If

  If Cards(Follower, i, 1) = TempFollowerPlayValue And Cards(Follower, i, 2) = TempFollowerPlaySuit Then

    For j = i To Trick
      Cards(Follower, j, 1) = Cards(Follower, j + 1, 1)
      Cards(Follower, j, 2) = Cards(Follower, j + 1, 2)
    Next j

    Cards(Follower, Trick, 1) = ""
    Cards(Follower, Trick, 2) = ""

  End If

End If
```

```

    If Cards(Follower2, i, 1) = TempFollower2PlayValue And Cards(Follower2, i, 2) = TempFollower2PlaySUIT Then
        For j = i To Trick
            Cards(Follower2, j, 1) = Cards(Follower2, j + 1, 1)
            Cards(Follower2, j, 2) = Cards(Follower2, j + 1, 2)
        Next j

        Cards(Follower2, Trick, 1) = ""
        Cards(Follower2, Trick, 2) = ""

    End If

    If Cards(Follower3, i, 1) = TempFollower3PlayValue And Cards(Follower3, i, 2) = TempFollower3PlaySUIT Then
        For j = i To Trick
            Cards(Follower3, j, 1) = Cards(Follower3, j + 1, 1)
            Cards(Follower3, j, 2) = Cards(Follower3, j + 1, 2)
        Next j

        Cards(Follower3, Trick, 1) = ""
        Cards(Follower3, Trick, 2) = ""

    End If

Next i

If TempFollowerPlaySUIT = Trump Then
    TempFollowerPlaySUIT = 6
Else
    If TempFollowerPlaySUIT = TempLeaderSUIT Then TempFollowerPlaySUIT = 5
End If

If TempFollower2PlaySUIT = Trump Then
    TempFollower2PlaySUIT = 6
Else
    If TempFollower2PlaySUIT = TempLeaderSUIT Then TempFollower2PlaySUIT = 5
End If

If TempFollower3PlaySUIT = Trump Then
    TempFollower3PlaySUIT = 6
Else
    If TempFollower3PlaySUIT = TempLeaderSUIT Then TempFollower3PlaySUIT = 5
End If

If TempLeaderSUIT = Trump Then
    TempLeaderSUIT = 6
Else
    TempLeaderSUIT = 5
End If

LeaderMan = 10 * TempLeaderSUIT + TempLeaderValue
FollowerMan = 10 * TempFollowerPlaySUIT + TempFollowerPlayValue
Follower2Man = 10 * TempFollower2PlaySUIT + TempFollower2PlayValue
Follower3Man = 10 * TempFollower3PlaySUIT + TempFollower3PlayValue

If LeaderMan > FollowerMan And LeaderMan > Follower2Man And LeaderMan > Follower3Man Then
    TrickWinner(Trick, Trump) = Leader
If FollowerMan > LeaderMan And FollowerMan > Follower2Man And FollowerMan > Follower3Man Then
    TrickWinner(Trick, Trump) = Follower
If Follower2Man > LeaderMan And Follower2Man > FollowerMan And Follower2Man > Follower3Man Then
    TrickWinner(Trick, Trump) = Follower2
If Follower3Man > LeaderMan And Follower3Man > Follower2Man And Follower3Man > FollowerMan Then
    TrickWinner(Trick, Trump) = Follower3

Leader = TrickWinner(Trick, Trump)

```

Module1 - 13

```
LeaderMan = 0
FollowerMan = 0
Follower2Man = 0
Follower3Man = 0
TempLeaderSuit = 0
TempLeaderValue = 0
TempFollowerPlaySuit = 0
TempFollowerPlayValue = 0
TempFollower2PlaySuit = 0
TempFollower2PlayValue = 0
TempFollower3PlaySuit = 0
TempFollower3PlayValue = 0
```

Next Trick

```
NumWinOne = 0
NumWinTwo = 0
NumWinThree = 0
NumWinFour = 0
NumWinFive = 0
```

For i = 1 To 5

```
Select Case TrickWinner(i, Trump)
```

```
Case 1
```

```
NumWinOne = NumWinOne + 1
```

```
Case 2
```

```
NumWinTwo = NumWinTwo + 1
```

```
Case 3
```

```
NumWinThree = NumWinThree + 1
```

```
Case 4
```

```
NumWinFour = NumWinFour + 1
```

```
End Select
```

Next i

```
TeamOneThreeWin = NumWinOne + NumWinThree
TeamTwoFourWin = NumWinTwo + NumWinFour
```

```
If TeamOneThreeWin > TeamTwoFourWin Then
```

```
HandWinner(Iteration, Trump) = 1
```

```
Else
```

```
HandWinner(Iteration, Trump) = 2
```

```
End If
```

Next Trump

Next Iteration

For i = 1 To 2

```
NumHeartTrumpWin(i) = 0
```

```
NumSpadeTrumpWin(i) = 0
```

```
NumDiamondTrumpWin(i) = 0
```

```
NumClubTrumpWin(i) = 0
```

Next i

For i = 1 To Iteration2

```
If HandWinner(i, 1) = 1 Then
```

```
NumHeartTrumpWin(1) = NumHeartTrumpWin(1) + 1
```

```
Else
```

```
NumHeartTrumpWin(2) = NumHeartTrumpWin(2) + 1
```

```
End If
```

Module1 - 14

```
If HandWinner(i, 2) = 1 Then
    NumSpadeTrumpWin(1) = NumSpadeTrumpWin(1) + 1
Else
    NumSpadeTrumpWin(2) = NumSpadeTrumpWin(2) + 1
End If

If HandWinner(i, 3) = 1 Then
    NumDiamondTrumpWin(1) = NumDiamondTrumpWin(1) + 1
Else
    NumDiamondTrumpWin(2) = NumDiamondTrumpWin(2) + 1
End If

If HandWinner(i, 4) = 1 Then
    NumClubTrumpWin(1) = NumClubTrumpWin(1) + 1
Else
    NumClubTrumpWin(2) = NumClubTrumpWin(2) + 1
End If
```

Next i

```
Worksheets("Sheet1").Cells(5, 2).Value = NumHeartTrumpWin(1)
Worksheets("Sheet1").Cells(6, 2).Value = NumSpadeTrumpWin(1)
Worksheets("Sheet1").Cells(7, 2).Value = NumDiamondTrumpWin(1)
Worksheets("Sheet1").Cells(8, 2).Value = NumClubTrumpWin(1)

Worksheets("Sheet1").Cells(5, 3).Value = NumHeartTrumpWin(2)
Worksheets("Sheet1").Cells(6, 3).Value = NumSpadeTrumpWin(2)
Worksheets("Sheet1").Cells(7, 3).Value = NumDiamondTrumpWin(2)
Worksheets("Sheet1").Cells(8, 3).Value = NumClubTrumpWin(2)

Worksheets("Sheet1").Cells(2, 1).Value = Iteration2
```

End Sub

References

Hogg, Robert V. and Allen T. Craig. Introduction of Mathematical Statistics,
Fifth Edition. New Jersey: Prentice Hall Inc., 1995.