2017

Neural Network Predictions of a Simulation-based Statistical and Graph Theoretic Study of the Board Game RISK

Jacob Munson

NEURAL NETWORK PREDICTIONS OF A SIMULATION-BASED STATISTICAL AND GRAPH THEORETIC STUDY OF THE BOARD GAME RISK

A Thesis
Presented to
the Faculty of the Department of Mathematics and Statistics
Murray State University
Murray, Kentucky

In Partial Fulfillment
of the Requirements for the Degree
of Master of Science

by Jacob M. Munson
December, 2017

Title

DATE APPROVED: _____

_____

Thesis Advisor

_____

Thesis Committee

_____

Thesis Committee

_____

Collegiate/School Graduate Coordinator

_____

Dean of the College/School

_____

University Graduate Coordinator

_____

Provost

Abstract

We translate the *RISK* board into a graph which undergoes updates as the game advances. The dissection of the game into a network model in discrete time is a novel approach to examining *RISK*. A review of the existing statistical findings of skirmishes in *RISK* is provided. The graphical changes are accompanied by an examination of the statistical properties of *RISK*. The game is modeled as a discrete time dynamic network graph, with the various features of the game modeled as properties of the network at a given time. As the network is computationally intensive to implement, results are produced by way of computer simulation. We propose three heuristic player strategies of increasing complexity, and demonstrate the effectiveness of each through a series of comparative simulations over a range of scalable values. The features are used to produce a prediction-oriented model based on these findings. The probability of a player win is modeled as a binary response to a single layer feed-forward neural network. We demonstrate the predictive power of our model as well as the performance increase of the player strategies. Recommendations for playing *RISK* well, based on these results, are given.

Acknowledgements

I want to thank the entire Murray State Department of Mathematics and Statistics for my time with them. I know completing both my B.S. and M.S. was a long time for many of them. Specifically, many thanks to my advisor, Dr. Christopher Mecklin, who took this project at my request and who also spent three semesters combing through related material with me. Appreciation goes out to my two other committee members, Drs. Elizabeth Donovan and Robert. G. Donnelly Jr. Thanks to Dr. Donovan for providing a brutal semester of graph theory. The contents of which have spilled into this work. Many thanks to Dr. Donnelly whose sage way of life serves to enlighten us all. Additional thanks to Dr. Justin Taylor for putting up with me for three semesters of analysis and not throwing chalk at me (yet).

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Purpose

This paper aims to develop the mathematical and statistical properties of diverse player strategies in the classic board game *RISK*. *RISK*, the game of global domination, is a strategic turn-by-turn game with up to six players. While many readers will be familiar with the style of the game, a brief overview of the rules and play will be provided.

The game is played on a predetermined board featuring 42 territories representing major geographical or political regions of the world. Between the 42 countries there are built-in connections. The connections are either physical borders that actually exist in geography or oceanic paths of travel as indicated on the *RISK* board. In all, there are 81 such connections. These connections serve as possible routes of travel for conquest. A RISK board is displayed in Figure 1.1 for illustrative purposes (Corporation, n.d.).

Figure 1.1: A RISK Board In Game

At the onset of the game, the 42 territories are divided between the number of players. The method for dividing the territories varies by player preference, but two main styles exist, both of which will be briefly outlined. First, in the game there exists a deck of cards referred to as *RISK* cards. The deck is representative of every territory and only represents such a territory once. Distributing the cards in a random fashion between the players will distribute all territories evenly between the players. The players then simply populate the territories indicated by the cards they have been dealt. Alternatively, the players can individually select the territories of their choosing in some alternating fashion with a randomized player getting first pick. This style helps to aid players in acquiring territories in regions they may prefer, but obviously offers no guarantee as another player may select the desired territory to deplete the remaining options in that particular region.

Once it is determined which territories belong to which players, the players begin populating the map with units. This is done by each player starting with a sum of 40 units to place. The units are placed in an alternating fashion player by player, typically placing units one unit at a time, with the first player being chosen randomly.

Now that all territories are occupied and supplied with troops, the game can begin. It is important to note that at no point in the game will a territory be absent of troops. Again, the player to go first is typically decided randomly. The first of the three phases of each turn is the *drafting phase*. The drafting phase is followed by the *attacking phase*, which is followed by the *reinforcement phase*. A detailed discussion of each phase will follow.

The drafting phase is characterized by a player receiving a number of units and placing them, typically strategically, on their own territories for later use. The number of units given to a player during the drafting phase is a function of how many territories are held and if that player controls any continents entirely. The player is given a base amount of units, $R = max\Big\{3, \lfloor v/3 \rfloor \Big\}$, based on possession of v territories. This guarantees a player receives a minimum of 3 units per turn, which will increase as the player controls more territories. These units can be placed on any territory held by the player. The units can be placed uniformly between all territories held, individually or in bulk on territories of their choosing, or some other scheme deemed valuable by the player. This is done without intervention or action by the opposing players.

The player may receive additional bonus units for either possession of an entire continent or for redemption of certain pairs of *RISK* cards. We will discuss the ways by which a player will receive units for possession of an entire continent in Continent Bonuses. The *RISK* cards present a layer of complexity which we will not address in this study for reasons of simplification. An analysis featuring the *RISK* cards could be repeated for all procedures we follow.

Once the player places all of their troops received at the onset of the draft phase, the player moves into the second phase of the turn, the *attacking phase*. The *attacking phase* is characterized by the player choosing to use excess troops on a territory in an

attempt to conquer adjacent territories held by another player. The attacking phase serves as an opportunity for the player to expand their control of the map, commit to strategic skirmishes, or delay action until the desired opportunity arises. Possible transitions during the attacking phase only include transitions onto territories held by opposing players; that is, the player cannot travel through their own territories. All of this is to say, the attacking phase entails two different players interacting, at the discretion of the player whose turn it is.

In the event that a player chooses to attack an enemy territory, the attacking player can use up to $A = n - 1$ attacking units from a single territory adjacent to the target territory, where n is the number of units on the territory initiating the attack. To initiate an attack, a minimum of one unit must remain occupying the territory from which the attack is initiated. The A units of the attacking player then engage in a skirmish with the defending units on the enemy held territory. We will explain the details of the skirmish between the attacking units, A, and defending units, D, in The Skirmish section.

The attacking player can initiate as many attacks as desired, from as many territories as desired, provided there are sufficient units available to attack. Upon completion of the attacking phase, the player moves into the third and final phase of the turn, the *Reinforcement Phase*.

The *Reinforcement Phase* allows the player to shift units from a territory held onto another territory held. For example, if the attacking player loses 5 units from an attacking force of 7 in conquering a territory, it may prove wise to move several units from another territory so as to reinforce the recently acquired position.

The *Reinforcement Phase* can be implemented in two ways. The first variation allows players to move units from any territory held to any other territory held. The second variation allows players to move units from any territory held to any other

territory through which a path of held territories can be traced. The second variation essentially requires a physical path to exist between the desired territories involved in the reinforcement. The player is typically free to implement only one reinforcement, while there exist variations of play in which a player can make as many reinforcements as desired. Once the *reinforcement phase* is complete, the turn ends. The next player undergoes their turn in precisely the same sequence of phases. The players alternate taking turns until a single player holds all territories.

1.2 The RISK Board

The *RISK* board represents a planar projection of Earth as a board game. As stated previously, the *RISK* board consists of 42 territories and 81 connections between them. The 42 territories represent various geographical regions on Earth, while the connections represent physical routes of travel by either land or sea. The territories can represent broad regions, such as The Middle East, smaller portions of large countries, such as the Western United States, or individual countries, such as Japan. The board is divided into 6 continents, which roughly correspond to the continents of Earth. The territories and the corresponding continents to which they belong are provided in Table 1.1.

Table 1.1: Territory List by Continent

Continent	**North America**	**South America**	**Europe**	**Africa**	**Asia**	**Australia**
Territory	Alaska	Argentina	Iceland	Egypt	Kamchatka	Indonesia
	Northwest Territory	Venezuela	Ukraine	North Africa	Japan	Western Australia
	Alberta	Brazil	Southern Europe	East Africa	Mongolia	New Guinea
	Ontario	Peru	Northern Europe	Madagascar	Irkutsk	Eastern Australia
	Western United States		Scandinavia	Congo	Yakutsk	
	Greenland		Great Britain	South Africa	China	
	Quebec		Western Europe		Siberia	
	Eastern United States				Siam	
	Central America				India	
					Kazakstan	
					Ural	
					Middle East	

1.2.1 Continent Bonuses

With regards to the *drafting phase*, players can receive additional units for complete control of continents. This section serves as a basis to understand which continents are likely to be the subjects of repeated invasion or persistent dispute. Table 1.1 listed the 42 territories and the continent to which each territory belongs. Table 1.2 includes three measures of each continent. The first item is the *continent bonus* of the associated continent. The *bonus* refers to the number of units a player will be rewarded for beginning their turn with control of the respective continent. Note that control of a continent during the opponent's turn does not guarantee that a player will receive a continent bonus. The second item is the number of territories in each continent, a value we refer to as the *size* of the continent. The third item in Table 1.2 is the count of paths into the continent, a value we individually refer to as an *entry*.

Table 1.2: Continent Information

Continent	**Bonus**	**Size**	**Entries**
N. America	5	9	3
S. America	2	4	2
Australia	2	4	1
Europe	5	7	4
Africa	3	6	3
Asia	7	12	5

1.3 The Skirmish

With a more thorough understanding of the *drafting phase*, we move to examine the individual skirmish featured in the *attacking phase* more thoroughly.

The details of the skirmish are straightforward. The attacker can roll up to three dice, but not more dice than number of attacking units. We let A denote the number of attacking units and let D denote the number of defending units. Both A and D are

subject to change as the skirmish evolves. For example, an attacking force of $A = 4$ units allows the attacker to roll a single die, two dice, or three dice. This decision is entirely up to the attacker. Conversely, if we assume the attacker has only $A = 2$ units for attacking, the player can roll a single die or two dice, but not three dice.

In a similarly defined situation, the defender can roll up to two dice, but not more dice than defending units, D. For example, if the defender has $D = 6$ units, then the defender is allowed to roll a single die or two dice. For a defending force of $D = 1$ unit, the defender can only roll a single die. Similar to the attacking situation, if a defender has options regarding the amount of dice rolled, this decision is made independently of the other players. Both players must choose to roll a non-zero number of die.

The two players then roll all of their dice simultaneously. The maximum value of the attacking dice is then matched with the maximum value of the defending dice. If applicable, the second highest attacking die is then matched with the remaining defending die. With these matchings, the attackers wins the individual roll if the value of the die is strictly larger than the corresponding defending die. The defender wins the individual roll if the value of the defending die is greater than or equal to the value of the attacking die; that is to say, the defender wins individual rolls which result in a tie. This comparison is done for both defending dice when applicable. Clearly, the attacker will have one unused die when rolling all three dice.

When the attacker wins an individual roll, the defender reduces the units on the defending territory by one. If two dice were rolled, the defender can lose up to two units per roll. Similarly, when the defender wins an individual roll, the attacker reduces the units of the attacking force by one. If the attacker rolled more than one die, the potential for loss is increased accordingly.

The attacker can choose to either:

1. stop attacking at any time; or
2. proceed in attacking until either of the following:
    a. the attacker has no remaining units to use for attacking ($A = 0$); or
    b. the defender has no remaining units for defending ($D = 0$).

In the event that the player chooses to stop attacking, the remaining attacking units simply retreat back to the territory from which they attacked. This option might be employed if a player suspects the odds of success have fallen out of their own favor or perhaps have another strategy in mind.

In the event that the player does not choose to retreat, the attack will continue until one player's force is defeated. If the attacking player loses all units in the attacking force, then the attack concludes with no change in possession of territories, though the units remaining on the defending territory are likely less than the units initially available. The attacker may choose to attack the same target with units from another territory, but can no longer attack from territory which has no available units to attack. If the defending player loses all units in the defending force, then the attack concludes with a change of possession of the territory which was being attacked. The attacking player now has the option to place any number of the remaining attacking force on the territory. The attacker must advance with at least one unit, as no territory can be unoccupied, however the attacker may choose to advance with a limited number of units so as to not leave another territory unnecessarily vulnerable. Clearly, understanding the probabilities associated with the skirmish will prove invaluable in pursuing mathematical underpinnings of the game. We produce a brief example in Table 1.3 similar to the example provided by Tan (1997):

Table 1.3: Skirmish Example

Turn	Number of Armies		Number of Dice Rolled		Outcome of the Roll		Number of Losses	
0	Attacker	Defender	Attacker	Defender	Attacker	Defender	Attacker	Defender
1	5	4	3	2	6,3,1	3,3	1	1
2	4	3	3	2	5,2,1	5,2	2	0
3	2	3	2	2	4,3	4,1	1	1
4	1	2	1	2	6	4,2	0	1
5	1	1	1	1	5	1	0	1
6	1	0						

Table 1.3 illustrates the relevant skirmish intermediate steps, including the attacking dice outnumbering the defending dice (and vice versa), the attacking die beating the defending die (and vice versa), and ultimately one player winning the skirmish.

Chapter 2

Model of the Game

Before outlining the contents of our model, we briefly review existing literature and provide key assumptions for our analysis. The purpose of this review is twofold: we aim to both implement existing procedures and compare the implications of our findings with the advice found in existing literature.

We first examine existing literature on the mathematical and statistical nature of the skirmishes in *RISK*. Primary sources of the information regarding skirmishes in *RISK* can be found in Tan (1997) and Osborne (2003). As we'll outline in the following section, the approaches by Tan and Osborne represent small-scale interactions in *RISK*. Our objective is to model these small-scale interactions as well as large-scale interactions later, with the aim of strategically implementing findings of the small-scale interactions in our analysis.

2.1 Markov Chain approximation to Skirmishes

This section is dedicated to reproducing the skirmish-focused results produced by Tan and Osborne as well as examining the recommendations set forth by each. The

discussion aims to demonstrate the ability to produce the probability of success of an attack given the number of attackers and defenders by way of a *Markov chain*. The primary questions at hand are laid out by Tan and repeated by Osborne:

> If you attack a territory with your armies, what is the probability that you will capture this territory? If you engage in a war, how many armies should you expect to lose depending on the number of armies your opponent has on that territory (Tan, 1997)?

To model this phenomena, we follow the notation of Osborne and Tan and let A be the number of attacking units and D be the number of defending units, as previously noted, at the beginning of an $A \times D$ skirmish. Furthermore, at the beginning the n^{th} turn, the state of the skirmish can be completely specified by providing the number of attacking units, a_n such that $0 \leq a_n \leq A$, and defending units d_n such that $0 \leq d_n \leq D$ (1997). Thus, we define the state of the skirmish to at the onset of the n^{th} turn to be

$$X_n = (a_n, d_n) \tag{2.1}$$

Clearly, from the definition of a_n, d_n, and X_n, the skirmish at turn $n = 0$ can be specified by the state $X_0 = (A, D)$, which corresponds to the beginning of the skirmish. Any skirmish represented by a state X_{n+1} can be probabilistically specified by providing the state X_n of the skirmish (1997). Then we have that our state space transition has the *Markov Property*:

$$\begin{aligned} &P[X_{n+1} = (a_{n+1}, d_{n+1}) | X_n, X_{n-1}, ..., X_1, X_0] \\ =&P[X_{n+1} = (a_{n+1}, d_{n+1}) | X_n = (a_n, d_n))] \end{aligned} \tag{2.2}$$

That is to say, the probability of success at any turn $n+1$ given all previous turns, depends only on turn n. Thus, the $A \times D$ skirmish can be modeled computationally as a markov chain given by $\{X_n : n = 0, 1, 2, ...\}$ (1997). The Markov chain can be described by correctly specifying the transition probabilities. Tan specifies the transition probabilities and assumes the maximum value of the attacker rolls and the maximum value of the defender rolls are independent values, as well as the second largest values of the attacker rolls and the remaining defender roll(Tan, 1997). Osborne corrects this specification made by Tan, noting that the assumed independence of the rolls is incorrect (Osborne, 2003). The reasoning behind specifying consecutive state spaces as dependent is straightforward, each state space is a direct result of the previous state which came about by either an initialization or a sequence of dice rolls. Osborne offers corrected calculations as well as revised strategy recommendations. The main strategic notion produced by Tan which warrants discussion is the following statement:

> That when both attacker and defender have the same number of armies, the probability that the attacker wins is below 50%. (This is because in the case of a draw, the defender wins.) (Tan, 1997)

While we find that this explanation is a drastic oversimplification, the correct specification of the transition probabilities produces a far different result. Given the aforementioned procedure for dice rolling and elimination of units, we model the Markov chain computationally to efficiently produce transition probabilitites for arbitrary values of A and D. Modeling the Markov chain computationally allows for insights into effective strategies as well as examination of many related phenomena to the outcome of a skirmish.

We are primarily interested in producing the probability of success of an attack. From the simulation of probabilities of success, we obtain other helpful measures such

a expected losses for both the attacker and defender (Osborne, 2003; Tan, 1997). We could similarly examine expected remaining units. As noted in our assumptions, to model the probability of success of an attack we assume the attacker commits to the attack and does not withdraw from the attack until there is a clear winner. Given the outlines for the dice rolling procedures, we produce the $A \times D$ matrix where each entry is the result of 2500 simulations in Table 2.1.

Table 2.1: Attacker Win Percent for $A \times D$ Skirmish

A \ D	1	2	3	4	5	6	7	8	9	10
1	**0.42**	0.11	0.03	0.01	0.00	0.00	0.00	0.00	0.00	0.00
2	**0.75**	**0.36**	0.20	0.09	0.05	0.02	0.01	0.00	0.00	0.00
3	**0.92**	**0.66**	**0.47**	0.32	0.21	0.13	0.08	0.05	0.03	0.02
4	**0.97**	**0.78**	**0.64**	**0.48**	0.36	0.25	0.18	0.12	0.09	0.06
5	**0.99**	**0.89**	**0.77**	**0.64**	**0.50**	0.40	0.30	0.23	0.16	0.12
6	**1.00**	**0.93**	**0.86**	**0.74**	**0.64**	**0.52**	0.42	0.33	0.26	0.19
7	**1.00**	**0.97**	**0.91**	**0.83**	**0.74**	**0.64**	**0.54**	0.45	0.36	0.29
8	**1.00**	**0.98**	**0.95**	**0.89**	**0.82**	**0.73**	**0.64**	**0.54**	0.46	0.38
9	**1.00**	**0.99**	**0.97**	**0.93**	**0.87**	**0.81**	**0.73**	**0.65**	**0.56**	0.48
10	**1.00**	**0.99**	**0.98**	**0.95**	**0.92**	**0.86**	**0.80**	**0.73**	**0.65**	**0.57**

A table of similar values can be found in Osborne (2003). The table can be used to determine the likelihood of attacker success, given values for both A and D. Simply locate the number of attacking units of interest down the leftmost column of the rows, the number of defending units as labeled across the columns, and the corresponding element of the table indicates the probability of attacker success for the given values. Additionally, values in which $A \geq D$ are boldfaced for readability and ease of reference in upcoming discussions. We have produced a 10×10 matrix for illustrative purposes in Table 2.1

Moreover, examination of the matrix reveals several trends. First, consideration of the main diagonal reveals an interesting pattern. The value corresponding to $A = D = 1$ is 0.42, which is greater than the next value in the diagonal, 0.36,

corresponding the $A = D = 2$. This result is curious in that the attacker has higher odds of success when initiating a 1×1 skirmish than a 2×2 skirmish, where the attacker clearly has more units (and thus more dice to roll). This trend reverses in the instance in which $A = D = 3$ and steadily increases as A becomes larger. Also interestingly, we see than when $A = D$ and $A \geq 5$, the probability of success for the attacker is greater than or equal to 0.5.

We are also interested in values of A such that $A \gg D$, as we suspect that the probability of success is very high for the attacker. Note however that $A \gg D$ being a "success" is limited entirely to the small-scale interaction of the particular skirmish of interest. Realistically, many instances in which $A \gg D$ will indicate a player is acting extremely defensively in general as the attacker has amassed significant units, and may not be indicative of a winning strategy. However, we proceed to examine the first column of the matrix in Table 2.2 for confirmation of our intuition.

Table 2.2: Attacker Win Percent for $A \times D$ Skirmish, First Column

D \ **A**	1	2	3	4	5	6	7	8	9	10
1	**0.42**	**0.75**	**0.92**	**0.97**	**0.99**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**

We see the probability of success increases substantially as we increase the value of A, while the probability of success for smaller attacking forces is still quite high. It becomes increasingly obvious that using more attackers drastically improves the odds of success of an attack. Effectively, drastically outnumbered defenses present an increased probability of success for the attacker. Similarly, we examine the first row of the matrix in Table 2.3 to examine an example in which $D \gg A$, where we expect a similar pattern to hold in favor of the defender.

Table 2.3: Attacker Win Percent for $A \times D$ Skirmish, First Row

A \ D	1	2	3	4	5	6	7	8	9	10
1	**0.42**	0.11	0.03	0.01	0.00	0.00	0.00	0.00	0.00	0.00

We see that the probability of success is very low for one attacker and decreases as we move to the right of the matrix as we add defenders. Effectively, when attackers are drastically outnumbered, the probability of success quickly decreases. Here we have mathematical validation to the intuition that drastically outnumbered attacks have an increased probability of failure.

We now produce a plot of the data in Figure 2.1 in which we label the x-axis according to the number of attacking units, A, while placing the data for D values 5, 15 and 30, as well as data for $A = D$. We produce a line through each of these sets of data for the count of defenders.

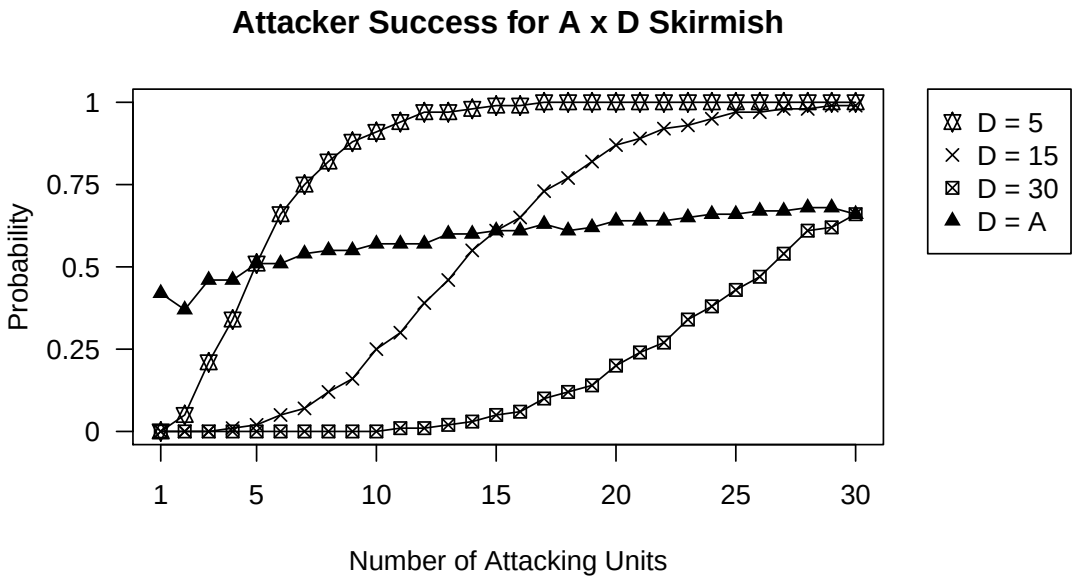

Figure 2.1: 2D Probability Plot for A x D Skirmishes

The trend for the line in Figure 2.1 $A = D$ indicates the trend we previously discussed, namely an initial dip for the 2×2 skirmish followed by steadily increasing values. Also, we see that instances in which D is very high result in a lower probability of success for the attacker. Similarly, instances in which D is lower correspond to higher probabilities of succuess for the attack.

We also visually examine the matrix by producing a surface plot in which the x-axis represents the number of attacking units, A, and the y-axis represents the number of defending units, D. The height of the plot in Figure 2.2 corresponds to the probability located at the intersection of the A^{th} row and D^{th} column of Table 2.1.

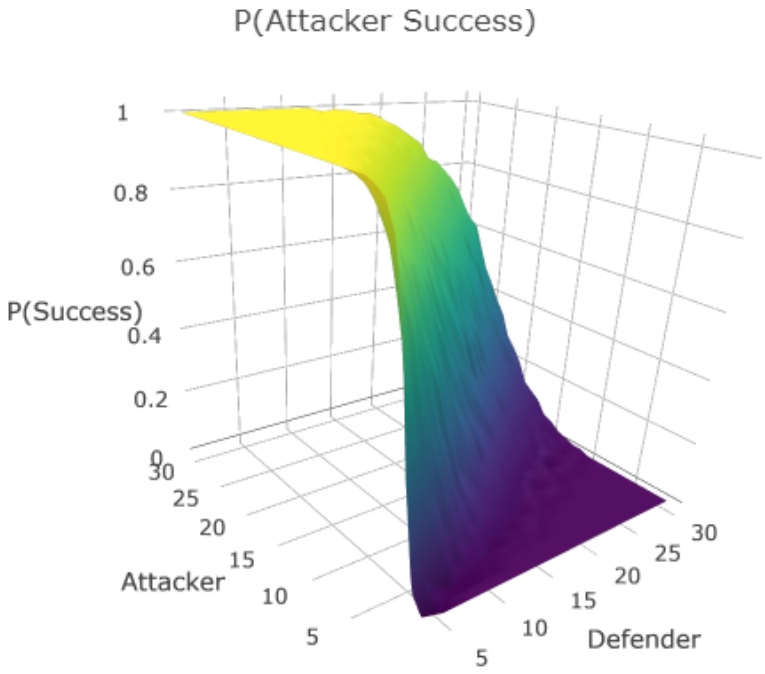

Figure 2.2: 3D Probability Plot for A x D Skirmishes

The probabilities for successful attack will help to serve as guidelines for attacking thresholds in subsequent strategy considerations. Given the probability of success for A and D, we move to examine the number of units that are expected to be lost in a skirmish by the attacker. We sample this distribution by simulation to produce the

results similar to the simulations for attacker win percent. We illustrate the findings in Table 2.4 in a similar $A \times D$ matrix which shows given an $A \times D$ skirmish, how many units are expected to be lost by the attacker.

Table 2.4: Expected Attacker Losses for $A \times D$ Skirmish

A \ D	1	2	3	4	5	6	7	8	9	10
1	0.59	0.89	0.97	0.99	1.00	1.00	1.00	1.00	1.00	1.00
2	0.67	1.41	1.66	1.86	1.92	1.97	1.98	1.99	2.00	2.00
3	0.57	1.40	1.90	2.30	2.54	2.71	2.82	2.88	2.93	2.96
4	0.54	1.52	2.07	2.67	3.02	3.34	3.53	3.69	3.79	3.86
5	0.52	1.51	2.19	2.85	3.38	3.79	4.12	4.37	4.55	4.68
6	0.52	1.54	2.23	3.01	3.59	4.16	4.56	4.92	5.19	5.40
7	0.52	1.53	2.27	3.09	3.77	4.37	4.91	5.36	5.72	6.02
8	0.52	1.54	2.28	3.14	3.85	4.55	5.16	5.70	6.16	6.53
9	0.52	1.54	2.30	3.18	3.91	4.66	5.36	5.94	6.50	6.96
10	0.52	1.54	2.30	3.18	3.98	4.74	5.47	6.15	6.76	7.31

From Table 2.4, we again take notice of an anticipated pattern. Consideration of the $A = 1$ and $D = 1$ element, we see the expected attacker loss is approximately 0.59. This is consistent with our previous finding that an attacker in such a scenario has a probability of success of approximately 0.42. Given minor rounding error, we could reason that since the probability of success is 0.42 in this instance, then $1 - .42$ is the probability of failure. The failure would correspond precisely to the attacker losing 1 units, which occurs with probability $1 - .42$.

Similar consideration of the first row of the matrix in Table 2.4 indicates for values of $D > 1$, the expected attacker loss approaches 1. Again, these results are consistent with our examination of the expected attacker win percentage, where the attacker is expected to lose with probability near 1.00 for greatly outnumbered attacks when $A = 1$. The first column of the matrix in Table 2.4 is equally informative. We see the expected attacker losses are roughly equal for $A \in \{1, 2, ..., 10\}$. This is consistent with our findings that the attacker win percent for attacking values $A \gg D$, when

$D = 1$ are almost a certain victory. Since minimal losses are experienced with low attacker values, we would not expect more attackers to be lost when adding to the attacking force only. We now produce a two-dimensional plot in Figure 2.3 of the matrix for various values of $A \times D$ with maximum values of $A = D = 30$ for a more broad view.

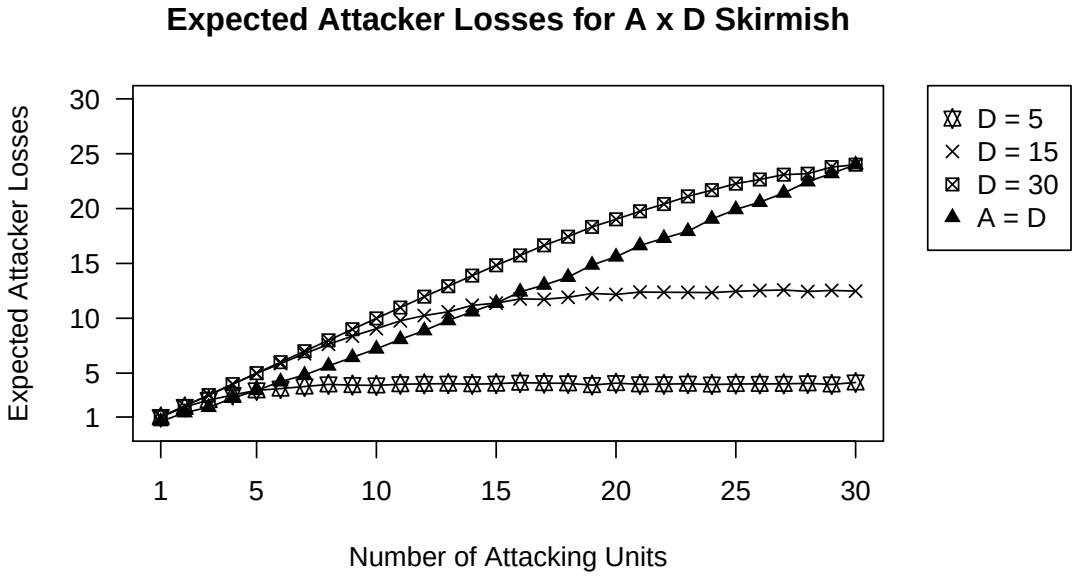

Figure 2.3: 2D Expected Attacker Losses for A x D Skirmish

The two-dimensional plot illustrates precisely the nature of the expected attacker losses that we could extract from further examination of the matrix. For further ease of viewing, we produce a surface plot in Figure 2.4 of the matrix.

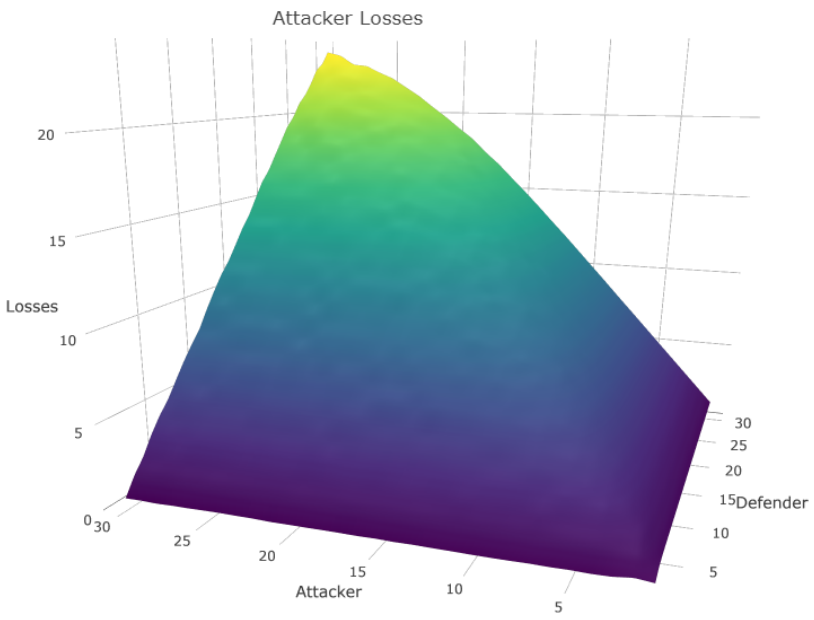

Figure 2.4: 3D Expected Attacker Losses for A x D Skirmish

The surface plot in Figure 2.4 is informative of the general behavior of attacker losses in $A \times D$ skirmishes. Notice, in instances of $A \times D$ in which the probability of success is very high for the attacker, we generally see correspondingly small values for expected losses. Conversely, in instances in which the probability of success was found to be very low, we see the expected losses are very high relative to the size of the attack, A.

We now examine the expected number of defending units to be lost in an $A \times D$ skirmish. We illustrate the findings in a similar $A \times D$ matrix in Table 2.5 which shows given an $A \times D$ skirmish how many units are expected to be lost by the defender, as opposed to the attacker.

Table 2.5: Expected Defender Losses for $A \times D$ Skirmish

A \ D	1	2	3	4	5	6	7	8	9	10
1	0.42	0.36	0.35	0.34	0.34	0.34	0.34	0.34	0.35	0.34
2	0.76	0.91	1.07	1.10	1.13	1.16	1.14	1.15	1.15	1.15
3	0.92	1.44	1.83	2.07	2.23	2.34	2.40	2.43	2.45	2.46
4	0.97	1.65	2.24	2.66	2.93	3.14	3.29	3.40	3.46	3.49
5	0.99	1.82	2.53	3.11	3.55	3.90	4.15	4.35	4.47	4.58
6	1.00	1.90	2.72	3.41	4.00	4.46	4.84	5.12	5.35	5.51
7	1.00	1.95	2.83	3.63	4.32	4.91	5.40	5.81	6.09	6.37
8	1.00	1.97	2.90	3.76	4.54	5.23	5.83	6.33	6.75	7.09
9	1.00	1.98	2.94	3.85	4.70	5.47	6.16	6.77	7.28	7.72
10	1.00	1.99	2.97	3.90	4.80	5.64	6.40	7.10	7.71	8.23

Similar to our examination of the expected attacker win percent and expected attacker losses, we can examine the matrix for verification of our intuition and consistency with our previous findings. The entry corresponding to $A = 1$ and $D = 1$ indicates an expected defender loss of 0.42 units. Recall, the probability of success for the attacker for such an event is 0.42, thus the defender losing all of the $D = 1$ units with probability 0.42. This result is consistent with previous findings. Also consider the first row of the matrix. We see the expected defender losses tend to stabilize across the first row, with very small losses. We expect few losses in these instances, as $D \gg A$ and $A = 1$ is almost a certain win for the defender, with very minimal losses. We now produce a two-dimensional plot of the matrix in Figure 2.5 for select values of the $A \times D$ skirmish with maximum values of $A = D = 30$ for a more broad view.

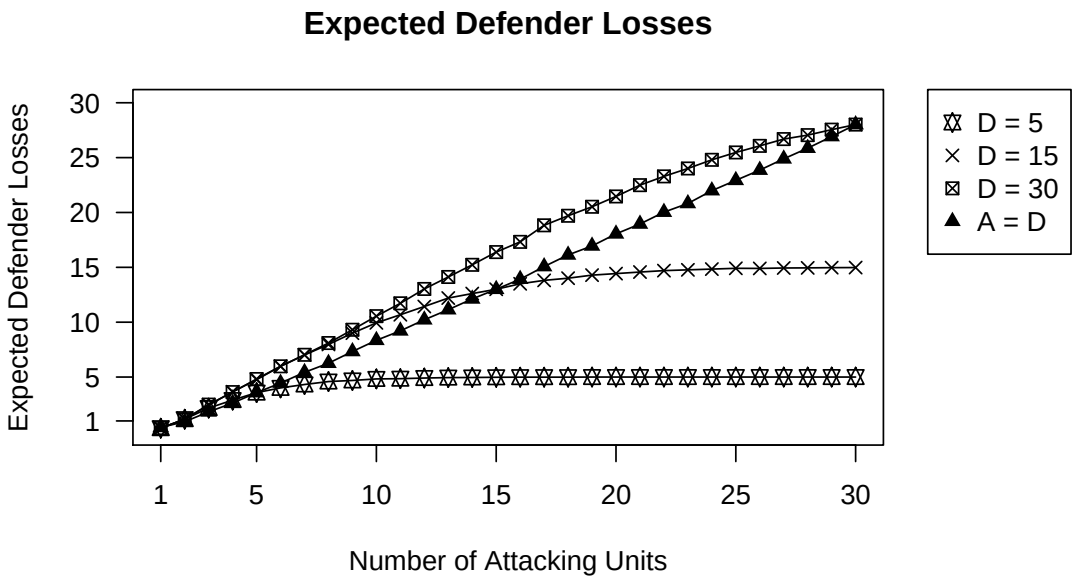

Figure 2.5: 2D Plot of Expected Defender Losses for A x D Skirmish

Similar to previous examinations, we produce a surface plot in Figure 2.6 of the associated matrix as well to further understand the *shape* of the data.

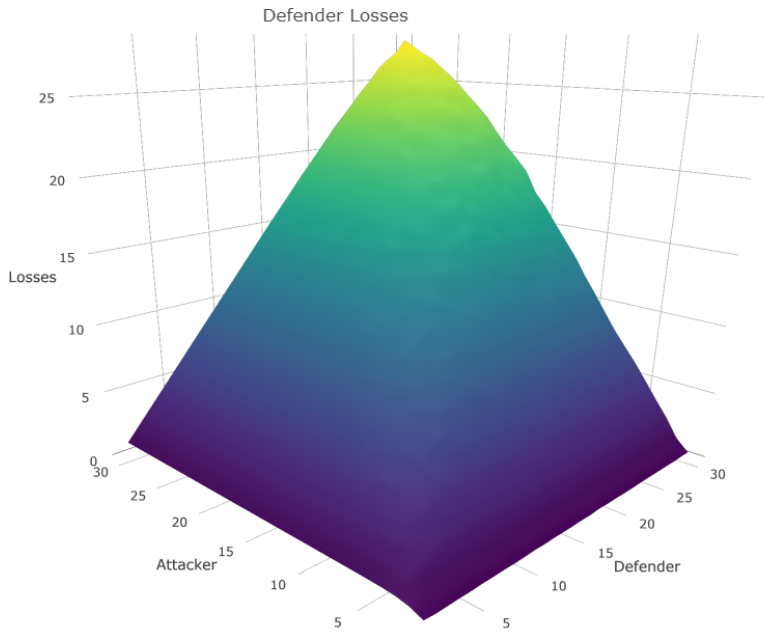

Figure 2.6: 3D Plot of Expected Defender Losses

Neither Tan (1997) nor Osborne (2003) take *hard* stances on their findings to dictate a recipe for appropriate play. Both authors suggest developing a personalized strategy based on experience and preference. Our aim is to do precisely this; to integrate the results of the probability of success findings from the initial Markov chain analysis in future decision making. However, as previously noted these findings are limited to the small-scale of an individual skirmish. We will find that many instances of "success" as found by individual skirmishes will be representative of an ineffective player strategy. Nevertheless, these are valuable considerations in the pursuit of strategy development and serve as a powerful explanatory tool.

2.2 Assumptions for Naive Analysis

We now list the pivotal assumptions we make in modeling the *RISK* game and provide justification for each. These assumptions aim to simplify the analysis while also

maintaining reasonable interactions. While some assumptions made prove unrealistic to how a human might act, we aim to build in measures to relax these assumptions once a firm foundation has been established. The strategies implemented in our analysis aim to provide a simple heuristic for game play. As suggested by Tan (1997) and Osborne (2003), a player will prove far more successful by developing a strategy fitting of them than attempting to rely entirely on an incomplete mathematical analysis. We echo the advice made by these authors; that is, a personalized strategy which takes mathematical findings into consideration when applicable under various assumptions and contrived situations will likely outperform strict adherence to our findings.

- The first simplifying assumption pertains to limiting the number of players in the game to two.
  - The limit of two players drastically reduces the number of possible moves a player can make while attacking, thus simplifying analysis.
  - Additionally, the areas to place defending units from either drafting or reinforcement will be drastically reduced.
- We assume each player rolls the maximum number of dice in skirmishes.
- We assume there is no limit to the amount of units a player can draft.
  - While there are material constraints in real life, we will assume there is a limitless supply of units in our RISK simulation(i.e. in real life there are a finite number of game pieces).
- The next simplifying assumption in the course of the game is the manner in which territories are given to the players.
  - We will assume the 42 territories are distributed randomly to the two players.

  - That is, each player will receive 21 randomly selected territories at the beginning of the game.

- Our next assumption details how player will distribute their drafted troops.
  - We assume players place all drafted units on territories which have an enemy territory adjacent to it.
  - We also assume that the players will evenly distribute their drafted units between such territories.
    - That is, drafted units will not be placed on territories surrounded by their own team.
  - This assumption ensures plenty of opportunities for attack will be available while also ensuring the player will not develop masses of units isolated from the other player.
- We will also assume, unless otherwise stated, that when an attack is available to a player, that attack will be made in full force.
  - Obviously, this assumption will lead to many haphazard engagements and situations in which an otherwise intelligent player would not enter.
  - We make this assumption as an aid to our assumption that units will be placed on outermost territories to ensure the game does not become stagnant.
  - Additionally, this assumption will be relaxed and modified heavily in our analysis, but serves as a very convenient basis.
- We assume that, in the event of a successful attack, a player advances with all possible units.
  - This assumption also serves the purpose of focusing the units toward the most active areas of the board.
- We will assume players do not implement the reinforcement phase of the turn to any extent.
  - This will help to provide areas of relaxation in more advanced analysis.

We define a player abiding by these assumptions to be a *naive player*. We will continually make modifications to these assumptions in an attempt to increase performance. The modifications will be based on statistical and graphical analysis of simulations of game of *RISK*. The naive player will serve as a baseline player from which we will measure improvement.

2.3 The Network Model

To observe these phenomena in mass quantity, we implement a *discrete time dynamic network model* built to provide specific outputs of interest. The advantages of building a computational model are vast. Foremost, we are able to produce far more data than we would otherwise be able to collect by observing or participating in live games. Essentially, we are able to produce as much data as our physical systems allow. Secondly, our results become reproducible. Should another individual attempt to implement or advance our procedures, our results will be consistent provided our algorithm is followed precisely.

The dynamic network model, M_t, is detailed below.

We define $M_t = \{G_0, G_t, A_t, L_t\}$ to be our model after t graphical updates, where $t \leq T \in \mathbb{N}$ and T is the time in which the game ends. Each instance of M_T corresponds to the completion of a single game of *RISK* as well as the information contained in $\{G_0, G_T, A_T, L_T\}$.

The components are defined as follows:

1. The *Underlying Graph*, G_0, refers to the *RISK* board translated into a graph. The edges that constitute the edge set of G_0 serve as a reference for which edges can be added back into graphs G_t as the vertices change color, a procedure outlined in the *Graphical Updates Section*. The vertices that make up the vertex

set of G_0 are the territories found on the *RISK* board.

2. The graph at time t, G_t, refers to the graphical embedding of the *RISK* board after t graphical updates.

3. The *List of Attributes*, A_t, of the model at time t includes properties of the *RISK game*, such as control of continents for each player, the draft count for each player, and starting player.

4. The *Unit List* at time t, L_t, is a $42 * 2$ matrix in which each row corresponds to a territory and each column represents a player.

2.3.1 The Underlying Graph

The *underlying graph* is represented as the *RISK* board translated into a graph. As outlined in the introduction, the *RISK* board features 42 territories and 81 connections between territories. We thus model the *RISK* board as a graph consisting of 42 vertices and 81 edges, where the vertices represent territories and the edges represent routes of travel on the RISK board. We will use this graph as an underlying graph from which we reference edges and vertices as our procedures make changes to our graph. We denote the original graph representing the unpopulated *RISK* board as G_0. One possible visualization of underlying graph of the RISK board is displayed below (Csardi & Nepusz, 2006).

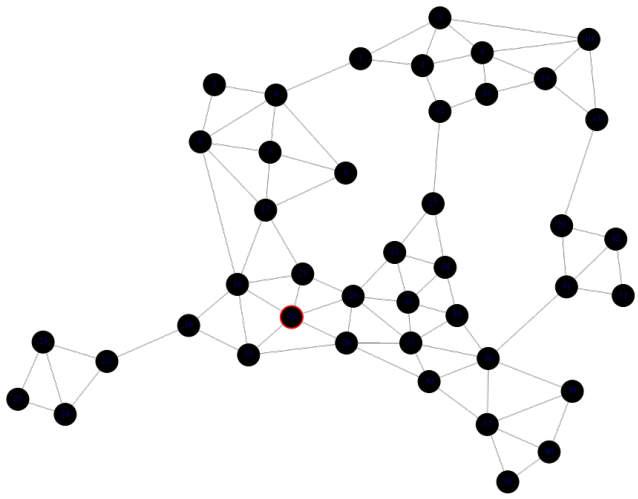

Figure 2.7: Underlying Graph

It is worth noting that the embedding of the above *underlying graph* is planar. While this is consistent with the *RISK* board representing a planar graph, the embedding of the *RISK* board at any time t need not be planar. For ease of visualization, planar embeddings will be implemented, though no part of our analysis requires such an embedding.

2.3.2 Graphical Updates, G_t

We now move to describe the nature of the changes we will make to G_0 in modeling the *RISK* game.

Most important of these changes is the deletion and adding of edges. We consider the color of vertices at time t, which is simply an indicator of player control. For example, if we let the vertices controlled by Player 0 to be red and the vertices controlled by Player 1 to be blue, the map will consist of blue and red vertices. Since *RISK* only allows for attacks from a territory held by a player to another territory held by the opposing player, edges between same color vertices are meaningless, since no travel is allowed on these edges. We will delete these edges as they arise. The

deletion of edges begins immediately upon the start of the game. Clearly, with every attacking phase, and indeed every attack, there is a possibility of edges being deleted. Additionally, when vertices become opposite in color, we will add the edges which exist in G_0, for if there is no such edge in G_0, then no edge is possible.

It is important to note that in the event of a successful attack, the attacking vertex and the recently conquered vertex will no longer share an edge, since the two vertices will be like in color. However, the conquered vertex underwent a color change. Given such a color change, there is a possibility that edges will need to be added from this vertex. We can think of the edges existing in the graph and essentially *turning on and off* based on the vertex colorings.

Each successful attacking event corresponds to a particular $t \in \mathbb{N}$ for count of events, t. An *update* on our graph is an events that inserts or deletes an edge existing in G_0 (Gross, Yellen, & Zhang, 2013). Since G_0 is undergoing updates, namely the addition and deletion of edges, we create a graph G_t for $t \in \mathbb{N}$ where t corresponds to a single successful attack. Thus, at the end of every successful attack we capture the structural changes from graph G_{t-1} to graph G_t.

Furthermore, a *dynamic graph* is a graph that undergoes updates of any form, including the addition and deletion of edges (Gross et al., 2013). A dynamic graph typically receives a sequence of updates, which we will index by t. In the sense that edges can be added or deleted, we consider the *RISK* game to occur on a dynamic graph. Each addition or deletion of an edge will correspond to a distinct time $t \in \mathbb{N}$ in a graph G_t. We then let our graph G_0 be a dynamic graph where G_t is as detailed above. It is important to note that as a dynamic graph we will neither add nor delete vertices for our implementation of *RISK*. Regardless of the current graphical embedding, our board will always contain the original 42 vertices from G_0. Simply put, deletion of a vertex would correspond to the deletion of a country on the *RISK*

board, which is an event that does not occur in the game.

Assuming our game terminates, a property we will discuss in *The Piece-Wise Ratio Strategy*, the end of our game will occur at some time T corresponding to a graph G_T in which all vertices are the same color. Given that edges are deleted when two territories share an edge in G_0, our final graph G_T will be completely disconnected and thus have 42 vertices and 0 edges. Thus, for any game of RISK represented graphically, the following ordered collection of graphs represents a game to completion:

$$\mathcal{C} = \{G_0, G_1, ..., G_t, ..., G_{T-1}, G_T\} \tag{2.3}$$

Each G_t can be both used to calculate graphical properties on an attack-by-attack basis and translated into an adjacency matrix. The graphical properties aim to reflect the structure of the graph. The structural changes will be shown to indicate clear trends as the game proceeds to completion. Thus, our model has the potential to examine the game on an attack-by-attack basis.

2.3.3 Attributes of the Game, A_t

As detailed above, each instance of an added or deleted edge will correspond to a particular $t \in \mathbb{N}$ for successful individual attacks. With each change in t, various game attributes will change. A brief description of each will follow. We consider the continent control of each player. Specifically, we count each turn a player controls a specific continent. Since there a six continents, each player will have six *continent control* values. For clarification, the continent control counts are incremented any time a player control a continent for an entire turn, the turns need not be sequential to accumulate. Simply, we are interested in determining to what extent the control of continents impact the outcome of the game.

We consider the starting player of the game. We'll denote the starting player as a simple binary in which St_0 indicates Player 0 started the game and St_1 indicates Player 1 started the game. We will track the starting player of each simulation and make determinations on the effect of starting the game in regards to the final outcome of the game. We are interesting in determining if starting the game, an event typically left to chance, has a significant impact on the outcome of the game.

We denote the draft at time t of each player as d_{0t} and d_{1t}, respectively. We use the maximum values of drafts for each player as our predictor the game by game basis, $D_0 = max\{d_{0t}|t \in \mathbb{N}\}$ and $D_1 = max\{d_{1t}|t \in \mathbb{N}\}$. In effect, this reduces the dimensions of the data for the draft count while still preserving the overall theme of the data. That is, a high max draft count will generally be indicative of a victory, while low max draft count will generally be indicative of a loss.

We track the number of turns in a game, T. Recall, G_T indicates a game which has terminated. Additionally, we give each player their own attack counter, t_{P0}, t_{P1}, respectively. Each turn can include multiple t_{P0} or t_{P1}, since a player is not limited to the number of attacks they can make in a turn, except by the availability of units.

2.3.4 The Unit List, L_t

The final structure of our model is the *Unit List*. The *Unit List* at time t, L_t, is a 42×2 matrix indicating a count of units on each territory by player. Each row of the *unit list* corresponds to a territory on the *RISK* board. The columns represent each player. The entry in the matrix corresponds to the units on a specific territory held by a certain player. The column of the player not holding the territory will have a zero in the corresponding row entry. Thus, each row the the unit list will have one positive integer and one zero.

A brief example is demonstrated below.

Table 2.6: Unit List Example

Player / Territory	Player 0	Player 1
Siam	5	0
Greenland	0	15
Ukraine	7	0

From above, we see Player 0 holds Siam with 5 units and Ukraine with 7 units. Also, notice Player 1 holds Greenland with 15 units. We see the opposing player has 0 units for their column when the other player holds the territory.

2.3.5 The Network Model Algorithm

Given all the described components of our network model, M_t, the algorithm for producing a game as $t \to T$ is as follows:

1. Territories are randomly and uniformed distributed between the two players on the graph G_0.
2. G_1 is produced to reflect deleted edges based on the territory distribution produced in (1).
3. The *RISK* Board is populated with the starting units given to the players.
4. The starting player is determined.
5. The Player determined in (4) begins turn.
6. Player drafts d units based on territory and continent possession.
7. The d units are placed uniformly on non-isolated vertices.
8. Player enters attacking phase.

9. Player determines possible attacks given distribution of units on possible attacking vertices.
10. Player attacks in accordance with appropriate assumptions.
11. For each unsuccessful attack, both attacking units and defending units on vertices are updated.
12. For each successful attack,
    - the vertex which was defending changes color;
    - units on vertices are updated;
    - edges are added and deleted appropriately; and
    - G_t is changed to reflect the vertex and edge changes
13. Player ends turn.
14. Opposite player performs steps (6-13).
15. Steps 6-14 are repeated until time T.
    - G_T is an empty graph on 42 vertices and all vertices are the same color.

Our algorithm produces a graph at each time t, game attributes at each time t, and a unit list at each time t, where t indicates individual graphical changes. Note, the algorithm given is with respect to our assumptions in Assumptions for Naive Analysis. The graph produced at time t can either be analyzed directly or converted to an adjacency matrix for analyis.

To analyze the events of our model output M_T, we consider two classes of predictors. The first class of predictors, the *game predictors*, have been outlined as contents of the game *attribute list*. These predictors aim to capture which player is favored to win as the game progresses. The second class, the *graphical predictors*, detail how the dynamic graph evolves into a completely disconnected graph at the

end of the game.

Notice both sets of predictors have limitations. The graphical predictors detail how the game is ending, but do not necessarily address which player is likely to win. Conversely, the game predictors are far more likely to address which player has the advantage, but fail to identify how the game graph is progressing structurally. Together, the two sets of predictors produce a more complete view of the game than either could individually offer.

2.3.5.1 Graphical Properties of G_t

This section is devoted to exploring some basic graphical properties of the graph which constitute our collection of graphs $\mathcal{C} = \{G_0, G_1, ..., G_t, ..., G_{T-1}, G_T\}$. As the game elapses in time, we will examine the changes in the properties at each discrete time $t \in \mathbb{N}$. Recall each t represents an attack by a player, which can result in a graphical change, but does not necessarily indicate a turn has elapsed (i.e. several player attacks can constitute one turn). However, since we are modeling the evolution of a *RISK* game as a discrete time dynamic graph, we have the ability to analyze each individual graph of the sequence of graphs in detail.

The graphical properties outlined will later serve a predictors in our statistcal modeling. Thus, we'll produce predictor values for each t of our collection of graphs $\mathcal{C}$. To illustrate the use and trends of our predictors, we produce a sample game of *RISK* purely for demonstrating these findings. The sample game produced a $42 \times (42 \cdot 80)$ matrix, and therefore consisted of 80 player attacks over the course of 17 turns. In the plots of the predictors displayed below, we additionally produce vertical dashed lines indicating the onset of turns 14 and 15 in each of the plots. We will see that many of the predictors display interesting behavior in this interval. First, we'll display the graph at each of these turns below (Csardi & Nepusz, 2006). The game graph at turn

14 is provided in Figure 2.8.

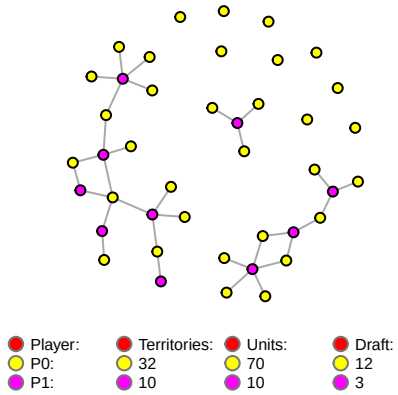

Figure 2.8: Sample Game at Turn 14

Figure 2.8 shows the game has already progressed a great deal by turn 14. Player 0 has control of 32 territories, has 70 units distributed on them, and is currently holding enough territories to draft 12 units per turn. Conversely, Player 1 has only 10 territories, with only 10 units distributed on them (hence, only one unit per territory), and is holding enough territories to draft the minimum draft amount of 3 territories.

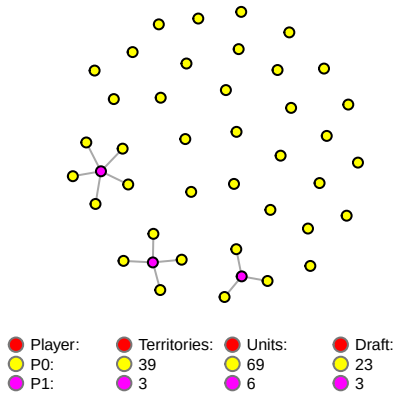

Figure 2.9: Sample Game at Turn 15

We see at the beginning of turn 15 given in Figure 2.9, the holdings of Player 1 have decreased substantially, from 10 territories to 3 territories. Note also that the player has 6 units now. In the previous turn, the player lost 7 territories (and therefore 7 units, since each territory had 1 unit), and then started the turn by drafting 3 units. Then from the previous turn, the player has $(10 - 7) + 3 = 6$ units. The draft given to Player 1 is unchanged at 3 units.

We proceed to explore the graphical measures we will implement while keeping the previous graphs in mind for reference. The first graphical measure we explore is the number of *cut vertices* in our graphs. Cut vertices are vertices whose removal increases the number of connected *components* in a graph (Gross et al., 2013). The components of a graph are given by maximally connected subgraphs. For example, in the graph G_0 the removal of the vertices corresponding to *Indonesia* or *Siam* would create a disconnected graph as shown in Figure 1.1. So, G_0 has two cut vertices. Note that as discussed, G_0 has received no removal of edges.

However, for graphs G_t which have been colored and received appropriate edge deletions, there will be many more cut vertices as the G_t graphs are in general more sparse than G_0. We include this measure to gauge the extent to which the graphs have essentially *fallen apart* with each successive time. Since the final graph G_T has no edges, G_T will have no cut vertices. Additionally, since G_0 has two cut vertices, we will see a spike in the count of cut vertices for times $0 < t < T$, after which the count of cut vertices will eventually fall to 0. Figure 2.10 is a plot of the count of cut vertices of a single simulation of a game of *RISK* (Csardi & Nepusz, 2006). With each graph of the graphical properties, in exclusion of the last properties, two dashed, horizonal lines are provided. The two lines indicate the beginning of the 14th and 15th turns, respectively. The aim of including these lines is to provided a sense in which the respective graphical property changes during the turns discussed in Figure 2.8 and Figure 2.9.

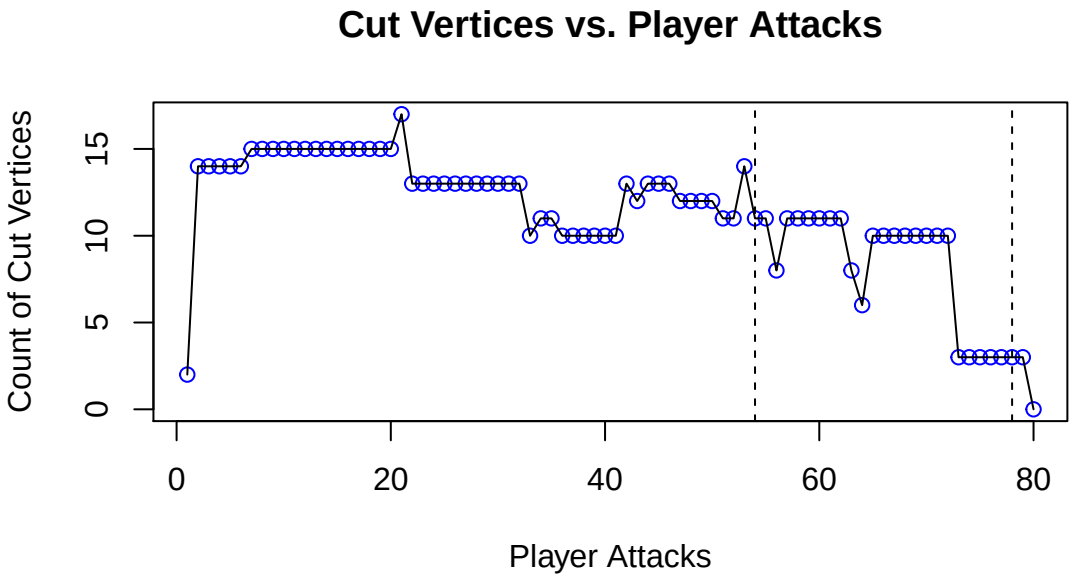

Figure 2.10: Cut Vertices of Sample Game

The next graphical measure we include in our analysis is the *average path length*

of the graph. The average path length is calculated by summing the lengths of the shortest paths between all pairs of vertices and dividing by the number of possible paths (Kolaczyk & Csárdi, 2014). The explicit equation is given by

$$\frac{\sum_{i \neq j} d(v_i, v_j)}{n(n-1)} \tag{2.4}$$

where $d(v_i, v_j)$ is the distance of the shortest path between vertices v_i and v_j. Clearly, for G_0 this value will achieve its maximum, as subsequent G_t graphs will by definition have less edges. Our aim in introducing the *average path length* as a predictor is to capture the average distance the two players share in splitting up the graph. Since only paths from opposite colors exist, the average path length will serve as a measure of the length of alternating colored vertices. Additionally, the average path length will decrease as $t \to T$, since G_T contains no paths. Note, however, that the *average path length* for an empty graph (e.g. G_T) is undefined. We simply truncate that instance and observe the defined values, since deleting the single data entry for an undefined *average path length* will not effect the overall trend of the data. Figure 2.11 is a plot of the average path length of our sample game of *RISK* (Csardi & Nepusz, 2006).

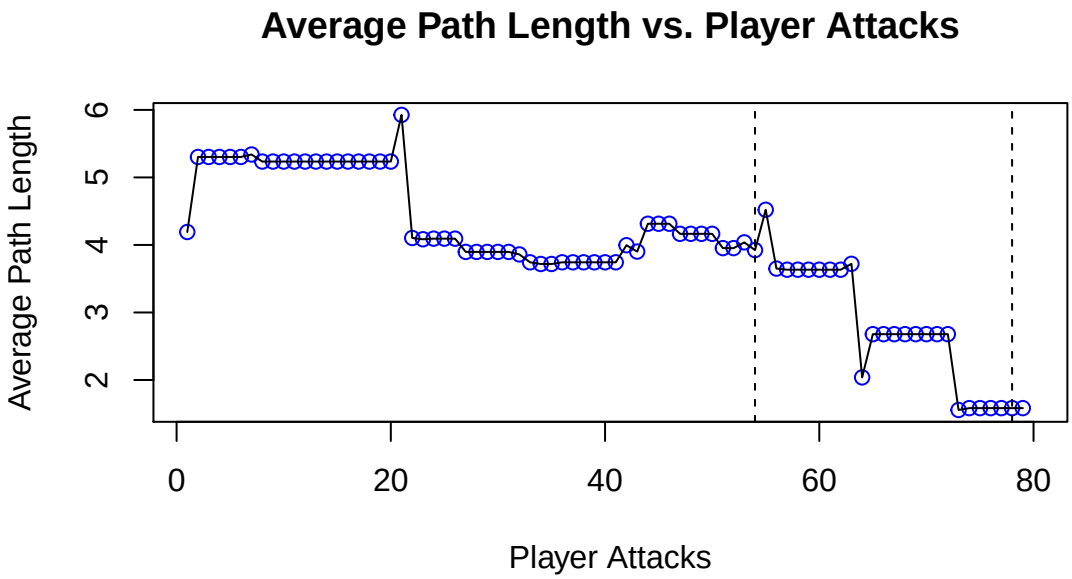

Figure 2.11: Average Path Length of Sample Game

A quick review of the plots of the *average path length* and *cut vertices* will likely convince the reader that these two quantities are highly correlated. The high correlation between these two quantities which will serve as predictors is an important point to notice in any statistical analysis. Such high correlation is frequently a cause for concern and will lead the data analyst to reexamine the use of at least one of the quantities. For the use of *average path length* and *cut vertices*, however, it is important to note that these two measures fundamentally express different values, and the plots will not necessarily look as similar as is shown in our example game.

We also examine the *maximum degree* of each of the graphs in the collection $\mathcal{C}$. The degree of a vertex is a basic structural property of a graph which is often used to establish a simple notion of importance of a vertex. Since our graphs are undirected and simple, the degree of the vertex is simply the number of edges incident on the vertex(Gross et al., 2013). To find the maximum degree of any graph in $\mathcal{C}$, simply find the degree of all vertices and the largest value in that set corresponds to the maximum

degree of the graph which can be expressed as

$$\Delta(G) = \max\{deg(v_i)|v_i \in V(G)\} \tag{2.5}$$

Examination of the graph for G_0 produced in Figure 2.7 shows a maximum degree of 6. We expect this value to trend downwards, since by definition G_T will have maximum degree of 0 and isolated vertices have degree 0(Gross et al., 2013). Again, the aim of including this measure is to capture how the graphs *fall apart* over time. We include a plot of the maximum degree for our sample game of *RISK* in Figure 2.12 (Csardi & Nepusz, 2006).

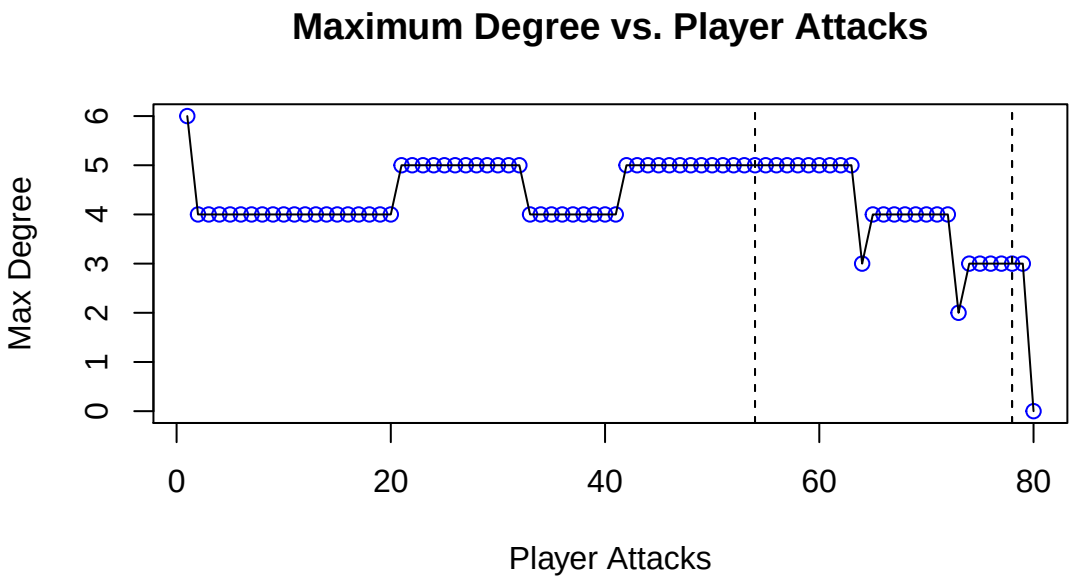

Figure 2.12: Maximum Degree of Sample Game

We next examine the *independence number* as a property of interest. The independence number of a graph is the maximum number of pairwise non-adjacent vertices in the graph (Gross et al., 2013). While this result can be found manually or computationally, we opt to find the independence number computationally given the

size of G_0 and the number of configurations of graphs in $\mathcal{C}$. To manually calculate the independence number of a graph G first select a vertex set $X \subseteq V(G)$ such that no edges exist between vertices in X. The largest cardinality of all possible sets X indicates the *independence number* of G. The independence number of G_0 is 16. This indicates that we can select 16 vertices from G_0 which will all be pairwise disjoint. Since G_t becomes more sparse as $t \to T$ the independence number will generally trend upwards as the game progresses. The final graph, G_T will have an independence number of 42 since each vertex is its own component in this graph. We use this measure to again capture the underlying changes in the graphical structure. The plot of the independence numbers for our sample game is provided in Figure 2.13 (Csardi & Nepusz, 2006).

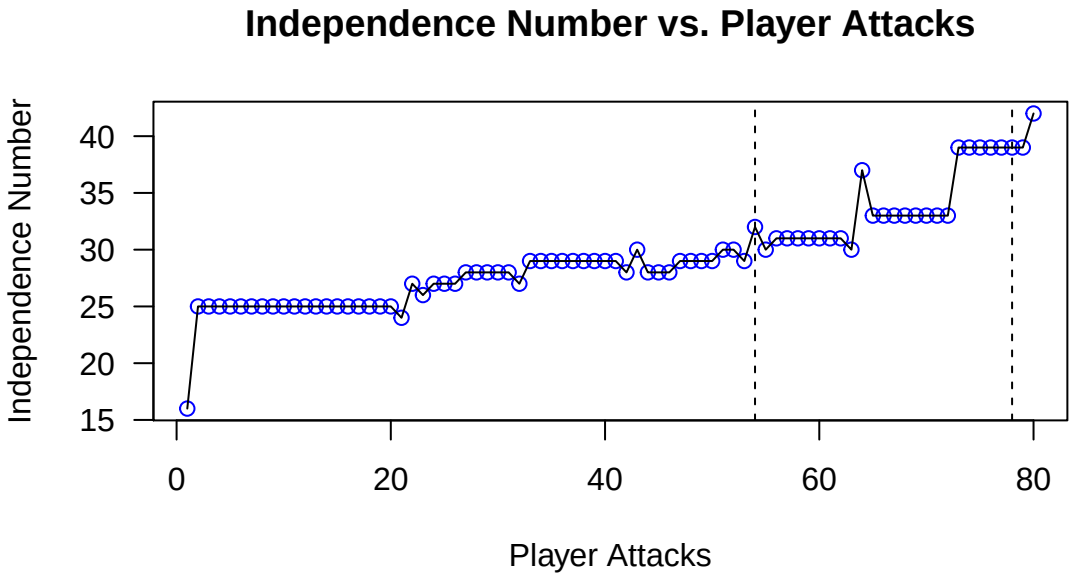

Figure 2.13: Independence Number of Sample Game

Another graphical measure we include is *graph density*. The graph density is the ratio of the number of edges in a given graph to the number of edges that could possibly exist in a complete graph of the same order. Since all graphs in $\mathcal{C}$ are simple

and undirected, the graph density is given by

$$\frac{2|E(G)|}{|V(G)|(|V(G)|-1)} \tag{2.6}$$

where $|E(G)|$ and $|V(G)|$ are the size of the edge set and vertex set of a graph G, respectively. Since G_0 will contain more edges than any other graph for $t > 0$, we expect the graph density to display a decreasing trend as $|E(G)|$ will trend downward and $|V(G)|$, and thus $|V(G)|(|V(G)|-1)$, will remain constant. Additionally, G_T has zero edges and will then have a graph density of zero. Figure 2.14 is a plot of the graph density for our sample game of *RISK* (Csardi & Nepusz, 2006).

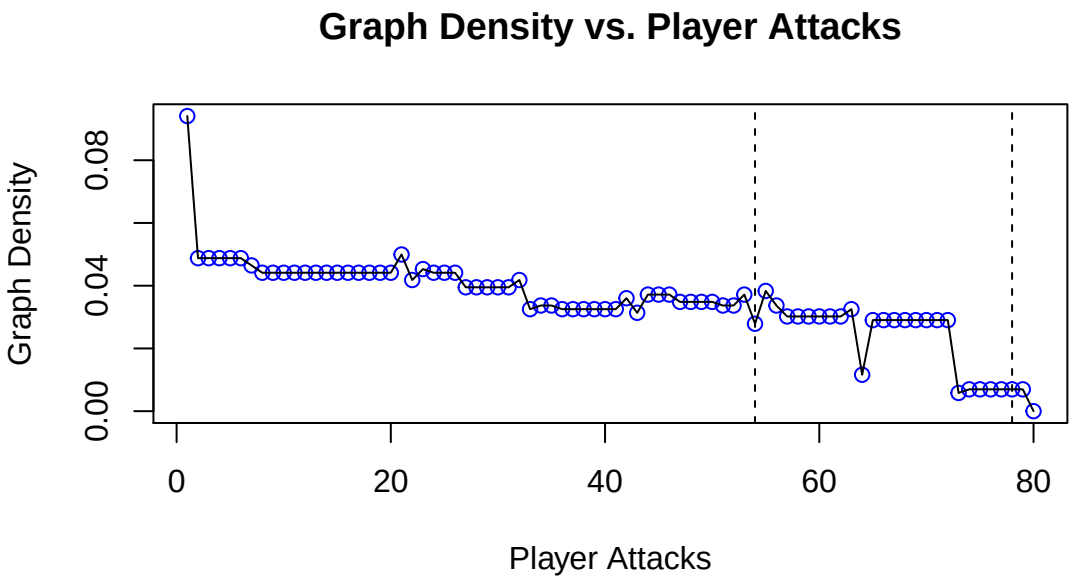

Figure 2.14: Graph Density of Sample Game

Next, we define a special class of graphs, the *4-Vertex Connected RISK Graphs*, which serve as *interesting* subgraphs for any graph in $\mathcal{C}$. The graphs of interest are, as the name suggests, connected graphs on 4 vertices. Given our model definition, these graphs given in Figure 2.15 will exhibit a proper coloring of only two colors. One

possible subgraph on four vertices is a cycle on four vertices ($\mathfrak{C}_4$), which will indicate instances in which each player has two territories which are themselves connected to two enemy territories connected to the original territories (Gross et al., 2013). The count of $\mathfrak{C}_4$ subgraphs will serve to measure how *mixed* the two players are on the board. Additionally, a path on four vertices (P_4) could exist which indicates a chain of length 4, oscillating between Player 0 and Player 1 (Gross et al., 2013). Again, instances of P_4 subgraphs will indicate the graph is still very *mixed* between the two players. The final possibility is a bipartite graph ($K_{1,3}$), which features one vertex essentially *surrounded* by vertices held by the opposite player. Examples of the three *4-Vertex Connected RISK Graphs* are provided in Figure 2.15.

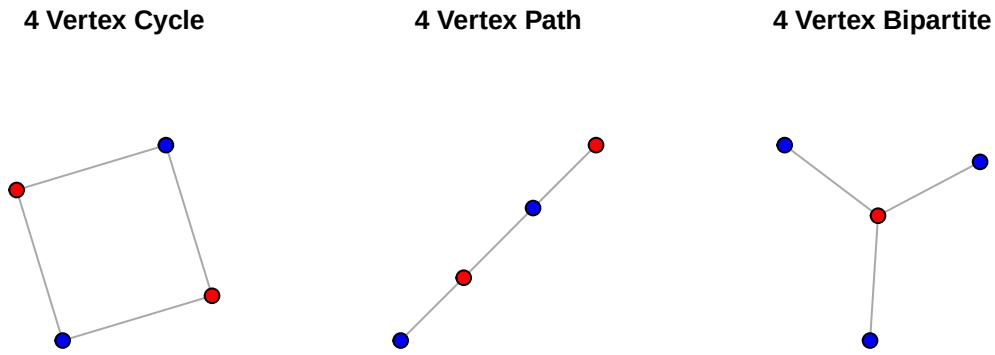

Figure 2.15: 4 Vertex Connected RISK Graphs with Possible Coloring

Naturally, as players get more established in areas of the graph or as the game progresses and one player's presence diminishes, the count of these graph will decrease. Initially, the graph G_0 has 455 instances of these subgraphs. Each graph G_t for $0 < t < T$ will have fewer of these subgraphs than G_0 since G_0 is the most dense graph in $\mathcal{C}$. G_T will have zero *4-Vertex Connected RISK Graphs* since it has zero edges and is completely disconnected. Below we plot the count of *4-Vertex Connected*

RISK Graphs of our sample game (Csardi & Nepusz, 2006).

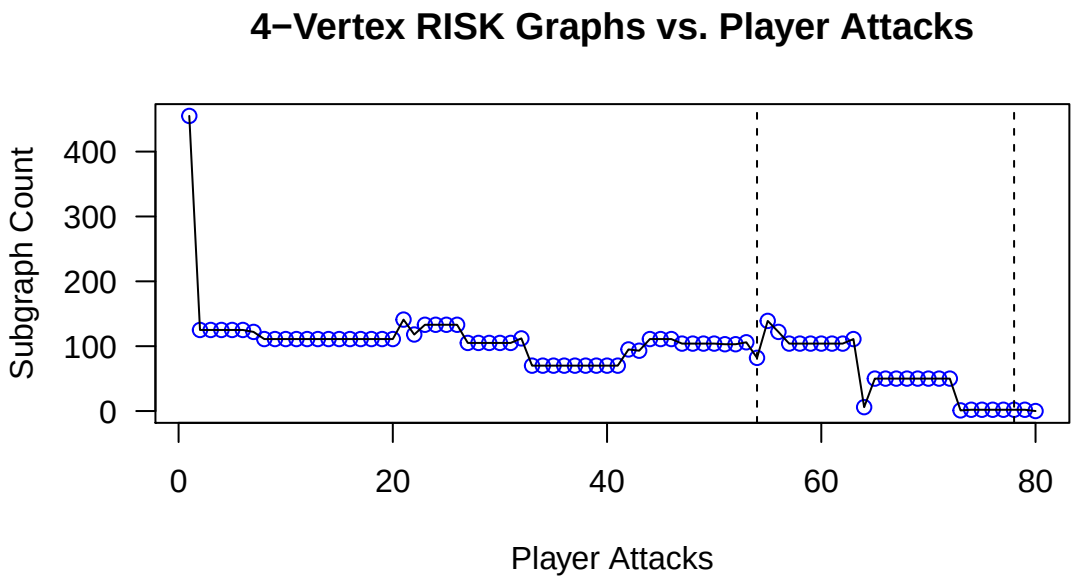

Figure 2.16: 4-Vertex RISK Graphs of Sample Game

We include a count of the number of components of our individual graphs; where a component is a maximally connected subgraph of the current graph G_t (Gross et al., 2013). Clearly, G_0 consists of one component, as any vertex can be reached from another(i.e., G_0 is connected). Graphs G_t typically consist of several components, as the deletion of edges makes certain vertices unreachable from others. In the end, G_T consists of 42 components. As a game elapses, we see an increasing trend from our graph having only one component, representing the connected G_0 graph, to the 42 components of G_T. Below we include a plot of the count of components for a single simulation of *RISK* (Csardi & Nepusz, 2006).

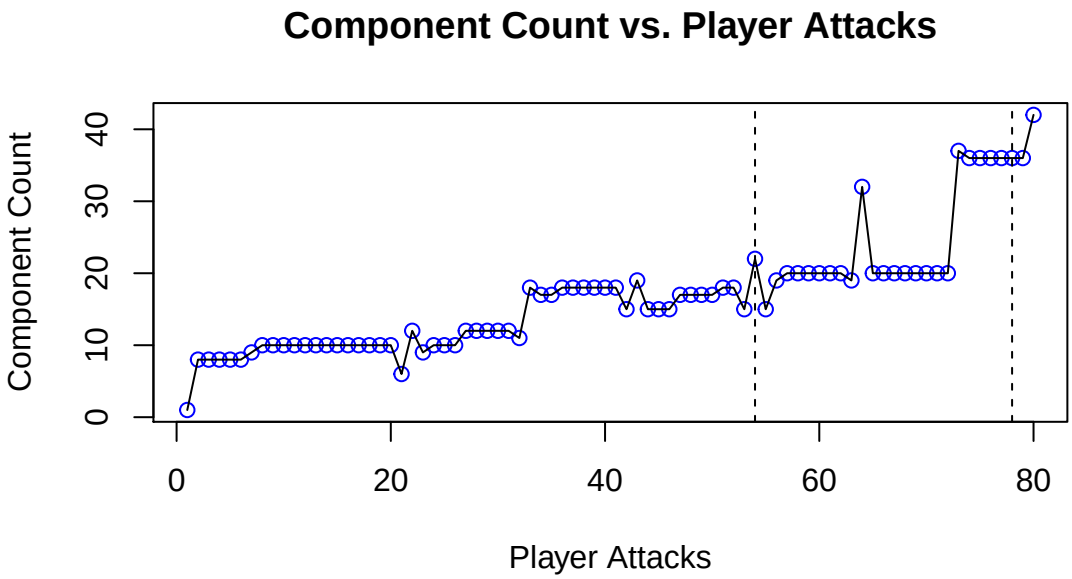

Figure 2.17: Component Count of Sample Game

We additionally include the diameter of our graphs in $\mathcal{C}$. The diameter of a graph G is defined by

$$max\{\epsilon(v)|v \in V(G)\} \tag{2.7}$$

where $\epsilon(v)$ is the maximum distance between v and any other vertex in G (Gross et al., 2013). Except for G_0, graphs G_t will feature diameters made up of territories held by alternating players. Note that for our model, a graph G_t can be specified by

$$G_t = \bigcup_{k=1}^{K} C_k = C_1 \cup C_2 \cup C_3 \cup ... \cup C_K \tag{2.8}$$

where C_k is a component of G_t. We consider the diameter of components of the graphs G_t for $0 < t \leq T$ separately since G_T is nearly always more than one component. Technically, from Equation 2.7 when G_t has more than one component we

have $diam(G_t) = \infty$. So, it is important to consider the components of G_t separately. We report the diameter of the graph G_t as the diameter of the component of G_t with the largest diameter given by

$$max\{max\{\epsilon(v)|v \in V(G_i)\}\} \tag{2.9}$$

Thus, stages in the game in which the two players are relatively separated, whether one player is nearly victorious or each player holds their own portion of the map, will have relatively short diameters. Games in which the territories of the two players are more mixed will feature larger diameters, which can be seen as paths of territories of alternating color. Furthermore, since G_T is all one color and features zero edges, all components of G_T will have diameter zero, thus the diameter will be zero. We then expect a downward trend in the diameter values. The plot of the diameter values for our sample game of *RISK* is displayed in Figure 2.18 (Csardi & Nepusz, 2006).

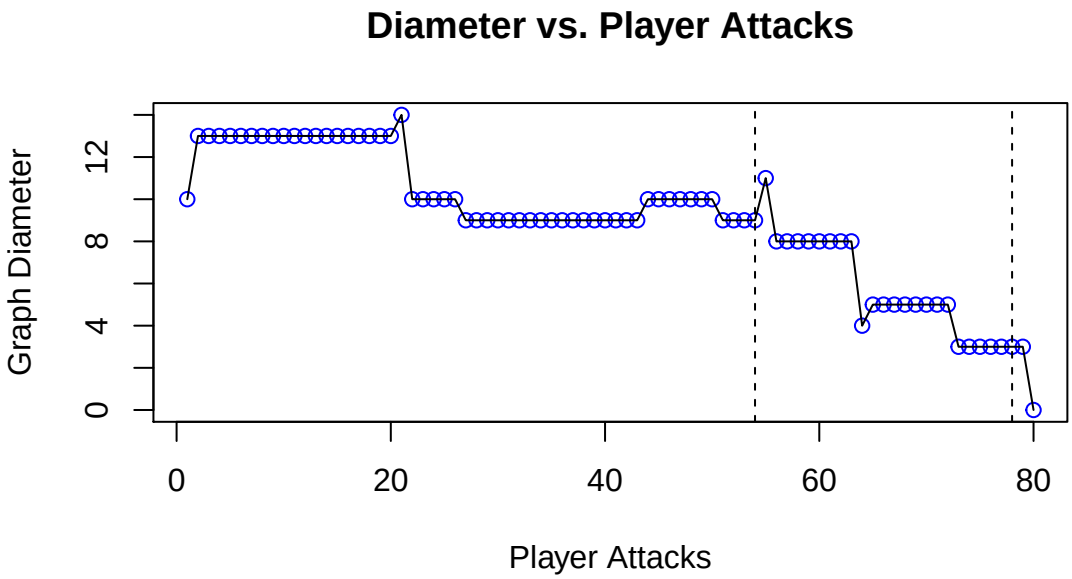

Figure 2.18: Diameter of Sample Game

The next two graphical predictors are graphical centrality measures. We include the *eigenvector* centrality and the *betweennesss* centrality. The essence of the centrality measures is to determine the relative *importance* of vertices in a graph based on how *central* a vertex is in regards to the entire graph. While intuitively simple, the two measures we include determine the importance of a vertex through mathematically different procedures.

Let's begin with the *eigenvector centrality* as a measure for our analysis. The *eigenvector centrality* scores of a graph require viewing the graph as its adjacency matrix. The interpretation of the eigenvectors is, in general, that higher scores correspond to vertices which are connected to many other vertices, which themselves are connected to many other vertices (Kolaczyk & Csárdi, 2014). Specifically, the individual eigenvector centrality meausures are given by solving

$$x_v = \frac{1}{\lambda} \sum_{u \in V(G)} a_{uv} x_u \tag{2.10}$$

where $x = [x_1, x_2, ..., x_n]$ and a_{ij} corresponds to the entries in the adjacency matrix, $A(G_t) = (a_{ij})$. The entry x_i corresponds to the eigenvector centrality of each vertex i (Gross et al., 2013). A plot of the eigenvalues of the eigenvector centrality scores for our sample game is displayed in Figure 2.19 (Csardi & Nepusz, 2006).

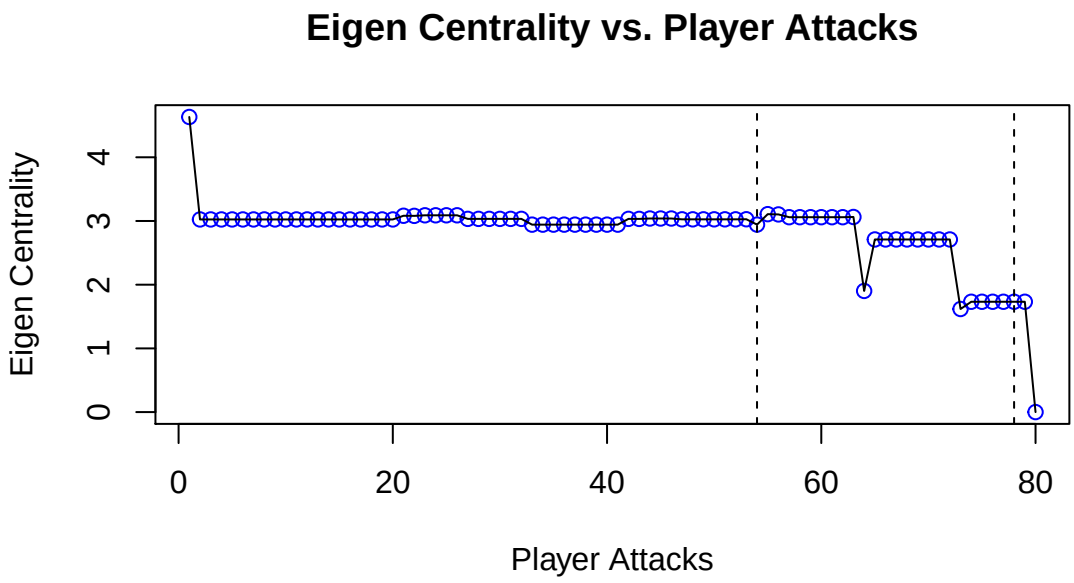

Figure 2.19: Eigenvector Centrality of Sample Game

The second centrality measure, the *betweenness* centrality, is the final measure of a graph for our analysis. The *betweenness* centrality can be applied to both edges and vertices, though in keeping with our stated desire to determine the centralities of vertices, we implement the vertex betweenness measure (Kolaczyk & Csárdi, 2014). The vertex betweenness is a measure which counts the number of shortest paths through a particular vertex (Gross et al., 2013). That is, for any vertex $v \in V(G)$, the *betweenness centrality* of v is given by

$$C_B(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}} \tag{2.11}$$

where $\sigma_{st}(v)$ is the count of shortest paths between distinct vertices s, t, passing through a vertex v(Gross et al., 2013). Since the graphs G_t are changing with each t, the betweenness measure is expected to change as the graph evolves. The betweenness measure for our sample game is displayed in Figure 2.19. Unlike the eigenvector centrality which reduces nicely to an eigenvalue(and all of our other graphical predictors which produce a single value), the *betweenness centrality* provides a value for each vertex. To simplify this data, we take the mean of the *betweenness measure* for each game. We find that the mean value is reasonably representative of the measure in general, and will produce results in the following section to substantiate this claim further. The plot of the betweenness centrality is shown in Figure 2.20 (Csardi & Nepusz, 2006).

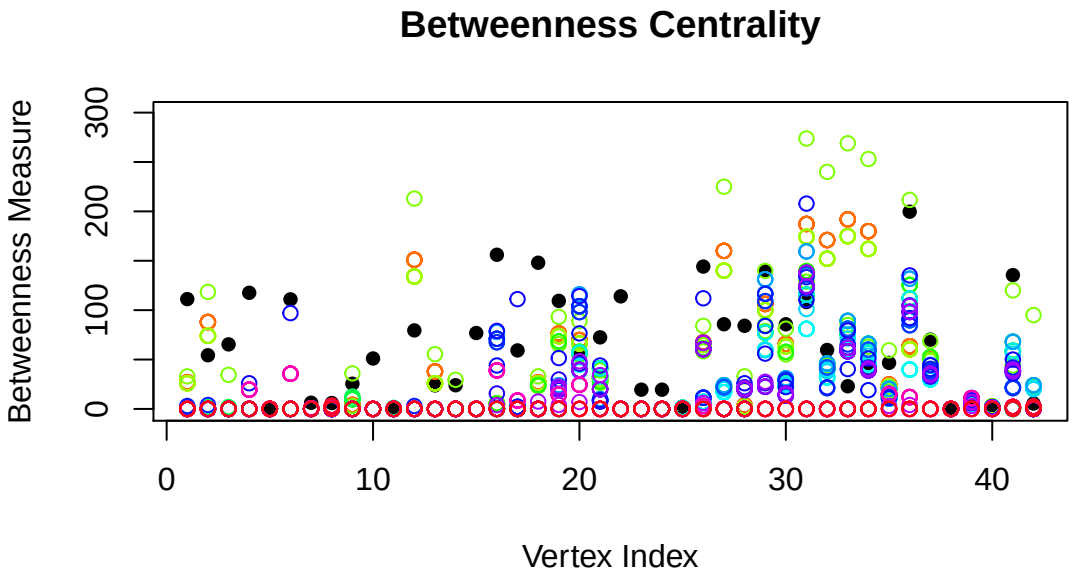

Figure 2.20: Betweenness Centrality of Sample Game

Since each graph, G_t, features changes from the previous G_{t-1} graph, the discussed graphical features will capture changes in graphical structure based on the action of the player. For example, a brief review of Player turns 1 to 20 on the predictor graphs show very small changes. This is indicative of a game being relatively uneventful at the beginning. We see that once a player amassed sufficient units, the game eventually *snowballed* into one player winning and we saw a corresponding change in the predictor plots. We are essentially measuring *how* and with what speed the game board graph is shifting towards an empty graph (i.e. $E(G) = \varnothing$). The implementation of predictors with the capacity to measure the effects of the graphical changes in addition to the first class of predictors, the game predictors, will help to provide a diverse view of the game.

With exception of the betweenness centrality, each of the remaining graphical predictors is further reduced in size by numerically integrating the data by way of Simpson's method (Curran, 2013). The objective here is to essential reduce each graph down into a single value. To be clear, we do sacrifice the ease of interpretation for the ease of modeling in this regard. For example, we present no interpretive value towards the integral of the eigenvector centrality curve for our dynamic graph. Such a value is unlikely to represent an intuitive notion of the status of the game to a casual observer as a game predictor such as Max Draft for Player 0 might. We suggest no interpretation for these values, we instead aim for a predictive capacity. The numerical integration of these parameters allows for a concise summary of the measures. Granted, the value from the integral will not be unique; that is, some other configuration of the parameters as time elapses could produce the same value (e.g., a long game with small values could produce a value similar to a short game with high values). However, from implementing these measures on many data sets, these graphs typically do not appear substantially different. Additionally, the benefit from implementing this procedure is vast, namely the reduction in the *size* of our data. For a game of T player attacks,

we produce T graphical measures and game measures, which require the storage and access of the associated adjacency matrix and the unit lists. For example, the sample game we've implemented is stored in $42 \times (42 \cdot 80)$ matrix, which occupies roughly 1.1Mb of disk space for a single game. With the aim of simulating tens of thousands of games, maintaining a value for each graphical change is impractical. Thus, we simply associate each game with an *integrated* value of our predictor curves. The aim is to create the association that the "amount" of area under the curve of the specific predictor is indicative of a certain graphical scenario which is itself indicative of the state of the game.

2.3.5.1.1 Review of Predictors

We provide a table of the *game predictors* and *graphical predictors* in Table 2.7. The table serves the purpose of summarizing the predictors as well as explicitly detailing the types of values that can be taken on by each as well as providing convenient abbreviations for each.

Table 2.7: Predictors

Predictor Classes			
Game Predictors		Graphical Predictors	
Predictors	Format	Predictors	Format
Player 0 Max Draft	$D_0 \in \mathbb{N}$	Graph Density Integral	$\int GD \in \mathbb{R}$
Player 1 Max Draft	$D_1 \in \mathbb{N}$	Component Count Integral	$\int CC \in \mathbb{R}$
Turn Count	$T \in \mathbb{N}$	Diameter Integral	$\int D \in \mathbb{R}$
Player 0 Attack Count	$A_0 \in \mathbb{N} \cup \{0\}$	Eigen Centrality Integral	$\int EC \in \mathbb{R}$
Player 1 Attack Count	$A_1 \in \mathbb{N} \cup \{0\}$	Independence Number Integral	$\int IN \in \mathbb{R}$
Starting Player	$St_0, St_1 \in \{0, 1\}$	Maximum Degree Integral	$\int MD \in \mathbb{R}$
Player 0 Africa Count	$Af_0 \in \mathbb{N} \cup \{0\}$	Cut Vertex Integral	$\int CV \in \mathbb{R}$
Player 1 Africa Count	$Af_1 \in \mathbb{N} \cup \{0\}$	Average Path Length Integral	$\int APL \in \mathbb{R}$
Player 0 Asia Count	$As_0 \in \mathbb{N} \cup \{0\}$	RISK Graphs Integral	$\int RG \in \mathbb{R}$
Player 1 Asia Count	$As_1 \in \mathbb{N} \cup \{0\}$	Mean Betweenness	$\mu(Bet) \in \mathbb{R}$
Player 0 Australia Count	$Au_0 \in \mathbb{N} \cup \{0\}$		
Player 1 Australia Count	$Au_1 \in \mathbb{N} \cup \{0\}$		
Player 0 Europe Count	$Eu_0 \in \mathbb{N} \cup \{0\}$		
Player 1 Europe Count	$Eu_1 \in \mathbb{N} \cup \{0\}$		
Player 0 N. America Count	$NA_0 \in \mathbb{N} \cup \{0\}$		
Player 1 N. America Count	$NA_1 \in \mathbb{N} \cup \{0\}$		
Player 0 S. America	$SA_0 \in \mathbb{N} \cup \{0\}$		
Player 1 S. America	$SA_1 \in \mathbb{N} \cup \{0\}$		

2.3.5.2 A Toy Example of the Algorithm

We'll now produce an example of our algorithm on a reduced scale featuring three non-descript vertices. The underlying graph is displayed in Figure 2.21 (Csardi & Nepusz, 2006). At each iteration of the algorithm, the graphical predictors can be produced. We forgo providing that data as we have previously demonstrated the predictors at full scale.

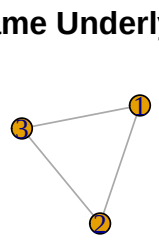

Figure 2.21: Underlying Graph of Example Game

1. Territories are randomly and uniformed distributed between the two players.

We next populate the underlying graph with players and units. We'll ignore the quantities of the draft units of for simplicity.

2. G_1 is produced to reflect deleted edges based on the territory distribution produced in (1).
3. The RISK Board is populated with the starting units given to the players.
4. The starting player is determined.
5. The Player determined in (4) begins turn.
6. Player drafts d units based on territory and continent possession.
7. The d units are placed uniformly on non-isolated vertices
8. Player enters attacking phase.
9. Player determines possible attacks given distribution of units on possible attacking vertices.
10. Player attacks in accordance with appropriate assumptions.
11. For each unsuccessful attack, both attacking units and defending units on vertices are updated.
12. For each successful attack,

- the vertex which was defending changes color;
- units on vertices are updated;
- edges are added and deleted appropriately; and
- G_t is changed to reflect the vertex and edge changes.

13. Player ends turn.

The graph featuring the above changes is displayed in Figure 2.22. Note, the edge found in the underlying graph between vertices 1 and 2 is no longer present, as the vertices share a color.

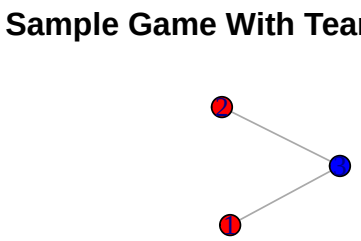

Figure 2.22: Example Game at t = 1

The associated unit list is given in Table 2.8. Notice, vertices 1 and 2 belong to Player 0 and there is not edge between vertices 1 and 2. Such an edge does exist in G_0, but it has been deleted due to the vertices belonging to the same player (i.e., they are the same color).

Table 2.8: Unit List of Example Game at t = 1

Player / Territory	Player 0	Player 1
1	10	0
2	1	0
3	0	5

14. Opposite player performs steps (6-13).

The implementation of steps 6-13 produce the graph in Figure 2.23.

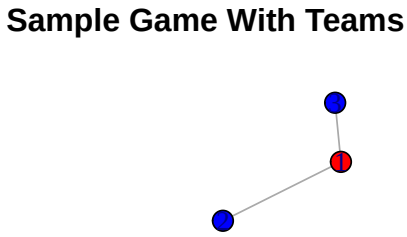

Figure 2.23: Example Game at t = 2

The graph in Figure 2.23 above corresponds to the unit list produced in Table 2.9.

Table 2.9: Unit List of Example Game at t = 2

Player / Territory	Player 0	Player 1
1	10	0
2	0	3
3	0	1

We now let Player 0 make successful transitions onto vertices 2 and 3 at time $t = 3$ and $t = T$.

14. Opposite player performs steps (6-13).
15. Steps 6-14 are repeated until time T.
    - G_T is fully disconnected and all vertices are the same color.

A successful transition onto vertex 3 is given in Figure 2.24.

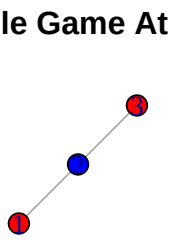

Figure 2.24: Example Game at t = 3

The unit list in Table 2.10 represents the change in units from the previous turn.

Table 2.10: Unit List of Example Game at t = 3

Player / Territory	Player 0	Player 1
1	1	0
2	0	3
3	7	0

We now allow Player 0 to make a successful transition onto vertex 2. Notice, given the successful transition made by the Player 0, we have deleted the edges which existed in the Example at $t = 1$. We now have a graph with zero edges in Figure 2.25, which corresponds to $t = T$.

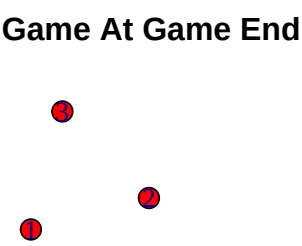

Figure 2.25: Example Game at t = T

The corresponding unit list is produced in Table 2.11.

Table 2.11: Unit List of Example Game at t = T

Player / Territory	Player 0	Player 1
1	1	0
2	5	0
3	1	0

Notice, the first column of Table 2.11 contains all non-zero entries, and therefore the second column contains all zero entries. This corresponds to Player 0 winning the game or, equivalently, holding units on all territories.

Chapter 3

Methodology: Strategy Implementations

3.1 Strategy & Simulations Overview

Given our computational model M_t, we proceed to implement simulations featuring various player strategies we define. The general procedure of our work is to define a strategy and perform simulations using that strategy. In all, we will define three distinct strategies and various sub-strategies. We also perform simulations in which various strategies are deliberately given continents to measure performance increases and changes both across and between strategies.

We perform 500 simulations for situations in which we do not require a high degree of accuracy. A sample size of 500 simulations produces a large standard error. However, we produce this data to simply observe trends, which does not require a detailed view of the data. Throughout our simulations, there are both portions where we are simply observing a trend or we desire to produce more data in areas of interest. If a trend of interest or area of focus is found, we simulate those portions more heavily.

On areas where we need more granularity or more certainty in our analysis, we increase the sample size considerably. For such simulations, we desire our simulations to estimate the true proportion p to the accuracy of the outcome of the game with standard error less than or equal to 0.01 from a sample of size n (Gelman & Hill, 2007). Given our binomial outcome, the standard error of the mean is:

$$\sqrt{(p(1-p)/n)} \tag{3.1}$$

Thus, for the population parameter we have

$$\sigma = \sqrt{(p(1-p)/n)} \tag{3.2}$$

Additionally, we can use $p = 0.5$ to produce an error bound as this value is lowest possible value of p(Gelman & Hill, 2007). Then by using $p = 0.5$ in our equation for standard error, we find

$$\begin{aligned} SE &= \sqrt{1/2(1-1/2)/n} \\ &= \sqrt{\frac{1}{4n}} \end{aligned} \tag{3.3}$$

By solving for n, we find $n = 1/4SE^2$. We allow for a standard error $SE = 0.01$, or 1%, and thus we require a sample size of $n = 2500$ simulations for a margin of error of approximately $2SE$, or $\pm 2\%$.

3.2 The Naive Strategy

With a thorough outline of the procedures of *RISK* and some rudimental findings on skirmishes, we define the *naive strategy* to be a strategy in which a player acts in full

accordance with assumptions in Assumptions for Naive Analysis Section. Strategically, the naive player places units uniformly and attacks as much as possible. The naive strategy essentially requires $A > 0$. For purposes of consistency, the naive strategy holds under any consideration in which $A \geq 0$, where $A = 0$ is essentially a null event. It is important to note that the naive strategy takes no information regarding the opposing player into account. This strategy will result in a high amount of activity, but not necessarily productive results.

3.2.1 Naive Strategy vs. Naive Strategy

A natural pursuit when defining the naive strategy is to explore the outcome of implementing two naive strategy players against each other. Thus, we use the naive strategy for both players of our game. We simulate 2500 games implementing the assumptions made for our analysis. Key of these assumptions is the layout of the game board at the onset of the game. In these simulations, neither player will deliberately receive a continent. Obviously, a player may still have a continent at any point in the game, including the beginning, given the random assignment of territories at the onset of the game. These simulations serve as a baseline for understanding the several effects. From these simulations, we find the win probability of Player 0 to be 0.497 and thus 0.503 for Player 1. With identical strategies and implementations, each player has a probability of winning of essentially 50%.

This result will help to understand later effects of interest including the impact of continent bonuses and the performance of the naive strategy under constrained situations. We will proceed with simulations in which players are given continent control in later sections.

3.2.2 Naive vs. Naive with Continent Control

This section serves the purpose of detailing the outcomes of simulations in which a naive player is given complete control of a continent at the onset of the game. One simulation entails an entire game of *RISK* from start to finish. In an attempt to understand the impact of continent bonuses on the outcome of the game, this preliminary analysis features a naive player opposing another naive player which is given a specified continent in each simulation. Specifically, these simulations center around Player 1 deliberately being given complete control of an individual continent at the onset of the game. Naturally, the player which is given a continent may not start the game with the continent. Simply put, Player 0 may start the game and invade any portion of the continent given to Player 1, as the starting player is still determined randomly in these simulations. The purpose of these simulations is to show that, while the continent bonus is not a guarantee, guaranteed possession of a continent at the onset of the game increases the odds of victory to varying degrees. We also briefly review relevant continent information in Table 3.1.

Table 3.1: Continent Information

Continent	Bonus	Size	Entries
N. America	5	9	3
S. America	2	4	2
Australia	2	4	1
Europe	5	7	4
Africa	3	6	3
Asia	7	12	5

For each continent set of simulations we simulate 2500 games. Again, this provides us with an acceptable standard error of 0.01. Table 3.2 features the results of these simulations in addition to a column dedicated to restates the results of the naive versus naive simulations previously produced. The first row of Table 3.2 is the probability of winning for Player 0, which is given no continents. The second row

of Table 3.2 is the probability of winning for Player 1, which is given the continent as indicated by the appropriate column. The third row of Table 3.2 indicates the advantage Player 1 has by assuming control of the relevant continent or layout.

Table 3.2: Naive Performance with Continents Given

Strategy \ Continent	None	Asia	N. America	Australia	S. America	Africa	Europe
Player 0	0.503	0.456	0.390	0.378	0.339	0.267	0.254
Player 1	0.497	0.544	0.610	0.622	0.661	0.733	0.746
Advantage	-0.006	0.088	0.22	0.244	0.322	0.466	0.492

We review these results in context for each continent below.

Asia:

Asia provides the largest continent bonus, so we might expect substantial impact on the outcome of the game. However, the probability of success for Player 1 starting with Asia is 0.544, which is slightly higher than the probability of success when no continent is given. There appears to be relatively little impact on the probability that a player wins when given Asia at the onset of the game.

We expect two factors to be at play here. First, Asia is a very large continent, featuring 12 territories. Maintaining such a large landmass at the beginning of a game will prove very difficult when resources are especially limited. Second, Asia has many entry points, as shown in the Table 3.1, from which a player could pose an attack in the event that Player 1 does not start the game, thus leading to the event that Player 1 did not actually receive the Asia continent bonus despite being given the continent. It seems likely that when Player 1 does not start the game, Player 1 will likely lose control of Asia and thus not receive the continent bonus on their first turn. However, this analysis should be not seem as a criticism on control of Asia as a method of increasing the probability of success. Though we found that onset control of Asia for a naive player does not substantially increase the odds of victory, it is likely that small incremental advantages similar to this finding produce substantial results for more

advanced players.

Australia:

We see drastically improved win percentage over the performance of the base scenario of no continents given. With a win probability of 0.622 for Player 1, this is far above the win percentage of Player 1 without a continent given. While the bonus for Australia is relatively small, a mere 2 units, it is important to note that Australia has only 4 territories and a single entry point. Thus, defending Australia, even in light of the naive strategy, is substantially less cumbersome than defending Asia, for example. Additionally, in the event that Player 1 loses control of Australia it is likely Player 1 will maintain *pockets* of units in the region, thus making recapture substantially easier.

North America:

For North America, we find the probability of a Player 1 win is 0.610. In light of our findings regarding Australia and Asia, this result is in line with expectations. North America is larger and more difficult to defend than Australia, so we expect the effect of the bonus to be less pronounced, which with a slightly smaller increase in win percentage, the effect is less pronounced. Also, North America is smaller and less difficult to defend than Asia, so we expect of the bonus to be more pronounced. Furthermore, with a bonus of 5 for North America, between the 2 and 7 for Australia and Asia, respectively, this win percentage is not particularly surprising.

South America:

We see when South America is given to Player 1 we have win probability of 0.661. For context, this probability is between that of Australia, at 0.62, and Africa, at 0.733. South America has two entries which is less than the three entries in North America, so from this perspective we expect the probability of winning to be higher

when given South America. Notably, however, South America is the same size as Australia and contains one more entry points, which would presumably make holding South America more difficult and decreasing the odds of success, a result which we do not find.

Africa:

We see Player 1 has a 0.733 probability of victory when given Africa at the onset of the game. While we do expect improvement upon our base case, this result is peculiar. Africa has 6 territories and 3 entry points. The six territories are more than that of say, Australia or South America, and the number of entry points is larger than those as well. Overall, Africa has a relatively moderate number of points of entry and a fairly small bonus associated with it. An attempt at understanding this result could be that Africa is relatively central in the map and will thus be involved in much dispute. Perhaps also that Africa is a relatively *deep* continent in that full invading Africa requires moving *down* the continent a substantial portion, that is to say that all of the entry points are at the *top* of Africa.

Europe:

Of all the continents, possession of Europe presents the highest odds of victory with a probability of 0.746. Recall, Europe is a relatively large continent with 7 territories and a bonus of 5 units. Furthermore, Europe has 4 entries which are relatively evenly spaced, one being from each of the other continents except Australia and South America. The high win probability associated with Europe is unexpected, much like we found with Africa. We suspect this result is an artifact of the position of Europe in the map. When a player holds Europe (and acts aggressively), it is unlikely that an opposing player will successfully hold a connected continent. Additionally, the continents not connected to Europe, which are South America and Australia, provide a bonus of only 2 units.

3.3 The Ratio Strategy, An Adjustable Approach

A natural progression of strategies upon the examination of the Markov chain approximation to skirmishes is to simply allow a player to attack when the units available to attack, A, are some fixed multiple, r higher than the number of units the opponent has available to defend, D. We will refer to this strategy as the *Ratio Strategy* and denote the corresponding ratio as r_P, where $P \in \{0, 1\}$ indicates Player 0 or Player 1, respectively. Note that this strategy does not rely on repeatedly sampling to produce the desired $A \times D$ probabilities. Thus, this procedure is far less computationally exhaustive than relying on continual Markov chain simulations for decision making, but relies on the reasoning found by the Markov chain approximation to skirmishes.

Furthermore, the naive strategy effectively allows attacks when $A > 0$ for some potential attacking force of size A. We reframe the strategy in terms of both A and D such that $A > 0 * D$. We then modify this strategy to only allow attacks when $A \geq r * D$, where r is some fixed ratio and D is the number of defending units on the opposing territory of interest. Clearly, when $r = 0$ the expression $A \geq r * D$ reduces to the naive strategy.

From the Markov chain approximation to skirmishes results, we see that when $A \geq D$ the odds of a successful attack are generally higher for sufficiently large A, as previously discussed. While this result is not particularly surprising, we will see a variety of outcomes based on possible selections of r. We will explore 81 possible values of r, where $r \in [0, 20]$ by increments of 0.25. This modifiable method will serve as a convenient basis for selecting when a player will attack under various restrictions.

While we do not expect the ratio strategy to be our final improvement in strategies or to accomplish a game theoretic optimal solution to *RISK*, as it is inherently exploitive in nature, it will serve as an important step toward understanding how to implement

a reasonably competent heuristic playing strategy. The objective of the following section is to explore how this range of values of r produce corresponding performance changes against a naive player.

3.3.1 Naive Strategy vs. Ratio Strategy

To further examine the performance of the heuristic ratio strategy, we simulate games of RISK in which the ratio strategy for Player 1 is implemented against the naive strategy for Player 0. The ratio strategy takes on values of r such that $r \in [0, 20]$ by increments of 0.25. The naive strategy is a special case of the ratio strategy with $r = 0$.

Each value of r is simulated for 500 games. Admittedly, the constraint of 500 simulations permits a great deal of error in the mean of these samples, an issue we address when examining the findings in more detail in the following section. Figure 3.1 displays the performance, by ratio value, for the ratio strategy player against a naive strategy player.

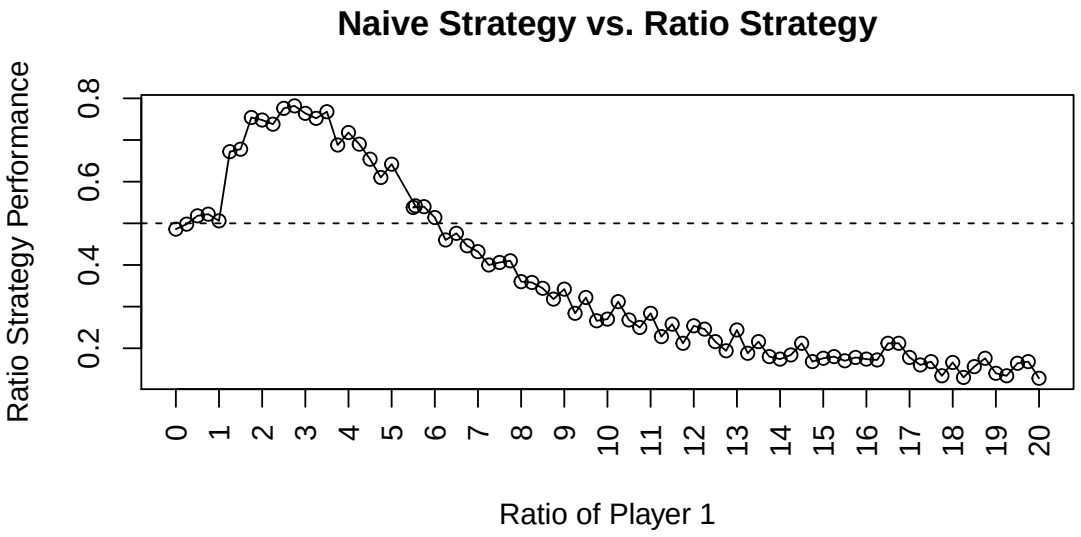

Figure 3.1: Naive Strategy vs. Ratio Strategy

The trend of the plot in Figure 3.1 is clear. For very low values of r, the performance of Player 1 and Player 0 is roughly equivalent. This result is expected, as low values of r indicate $A \geq r * D \approx 0 * D$, which reduces to the naive strategy. Naturally, as r increases in values near 0, the performance changes. At roughly a value of $r_1 = 0$ we see a spike in the performance of the ratio strategy. The performance continues to increase until we reach $r_1 \approx 3$. Following this peak, the performance drops at a rate similar to the increase until $r_1 \approx 5$. For $r_1 > 5$, the performance continues to drop, albeit at a slower rate. We display data until $r_1 = 20$, at which points the probability of success for the ratio strategy has fallen to nearly 0.10.

From Figure 3.1, we can see that overly aggressive play against a naive player effectively becomes a game of naive strategy versus naive strategy. Alternatively, overly defensive play, indicated by large values of r_1, essentially becomes a game in which the ratio player performs poorly, likely the result of the naive player making steady gains while the ratio player waits "too long" until an unrealistic criterion is met. Between these two situations, we have intermedate values of r_1 in which the

performance of the ratio player outperforms the naive player with a win probability reaching as high as nearly 80%. These intermediate values of r_1 are thus of particular interest as they would likely provide the player a good heuristic strategy against an average or weak player. We devote the next section to examining these values more closely.

3.3.1.1 Ratio Strategy Optimization in Ratio Strategy vs. Naive Strategy

In this section we aim to provide a more granular look at the range of ratio values which appear to contain a maximum for the performance of Player 1 implementing the ratio strategy against Player 0 implementing the naive strategy. To achieve this more granular view, we examine the range of r_1 values of $[0.95, 4.5]$ based on our examination of the Naive Strategy vs. Ratio Strategy. Furthermore, we increment values of r_1 by 0.05, as opposed to increments of 0.25 on the range of r_1 values in $[0, 20]$ in our previous examinination of the two players interacting.

Additionally, we perform 2500 simulations for each ratio value, instead of the prior 500 simulations for the global outlook, to ensure we reduce error in our sample sets, as previously discussed. Below we display a plot of the data collected through these simulations.

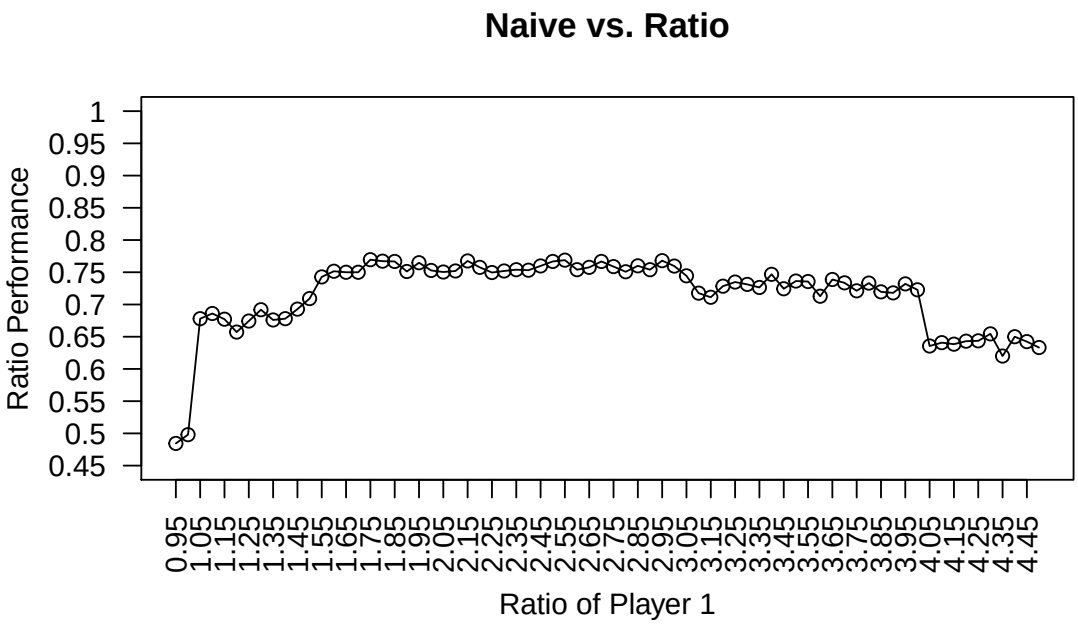

Figure 3.2: Naive vs. Ratio, Optimized by Perspective of Ratio Player 1

Figure 3.2 confirms our initial thoughts that this range of ratio values contains a maximum. Admittedly, the natural variance in our data does not allow for extreme values to be extracted with certainty, nor is that our intention. The collected data serves as a guideline for defining various player strategies within the ratio strategy. We thus define the three ratio strategy categories as follows:

Table 3.3: Ratio Strategy Styles

Player Strategy	Ratio Range	Probability of Win
Aggressive	[0, 1.5]	≤ 0.70
Balanced	(1.5, 4)	[0.70, 0.80]
Defensive	[4, ∞)	≤ 0.70

Given these definitions, we will further explore the performance of the ratio strategy by again assessing performance of a player when continents are provided to the player. The next section will focus on the performance of the ratio strategy for $r_1 = 2.5$, which will be termed as the *Optimized Ratio Strategy* when a naive

strategy opponent is given continent control as was simulated in the following section. Additionally, we will explore alternate strategies and assess the definitions of these categories of players from various perspective.

3.3.1.2 Optimized Ratio Strategy vs. Naive Strategy Given Continents

We now simulate the ratio strategy at $r_1 = 2.5$ against the naive strategy ($r_0 = 0$) given continent control. These simulations are similar to the previous simulations featuring the naive strategy against the naive strategy with continents. We found above that the ratio strategy at $r_1 = 2.5$ wins approximately 77% of games against a naive strategy player when no continents are deliberately given.

We are interested in finding where the performance of the ratio strategy player lies between the simulations of how the naive player performs against another naive player with continents, and how the ratio strategy player performs against the naive player without continents. A summary of these simulations is presented in Table 3.4 .

Table 3.4: Optimized Ratio Strategy vs. Naive Strategy with Given Continent

Strategy \ Continent	None	Asia	N. America	Australia	S. America	Africa	Europe
Optimized Ratio	0.769	0.58	0.519	0.652	0.626	0.521	0.386
Naive	0.231	0.42	0.481	0.348	0.374	0.479	0.614
Difference	0.538	0.16	0.038	0.304	0.250	0.042	-0.228

Recall that the optimized ratio strategy has a win percentage of approximately 77% against the naive strategy given no continents. From Table 3.4 we see the possession of continents certainly did aid the naive player. However, for control of Asia, North America, Australia, South America, and Africa given individually, the optimized ratio player still outperformed the naive player. Obviously, the performance of the optimized ratio player dropped significantly below the 77% win percentage it previously held. Only the possession on Europe as the continent given was enough to increase the win percentage of the naive player above a probability of 0.5. In

this instance, the optimized ratio player had a drastic performance decrease, with a percentage of *only* 39%.

3.3.2 Ratio Strategy vs. Ratio Strategy

The next step in evaluating our proposed strategies is to *cross* the performance of the ratio strategy against another player implementing the ratio strategy. We define a *cross* of the performance of two ratios r_0 and r_1 to be $Pr(Win|\{P_o(r_0), P_1(r_1)\})$, where $P_i(r_i)$ represents Player i implementing the ratio strategy with ratio r_i. Note that for an $M \times N$ matrix of M ordered r_0 values and N ordered r_1 values, the $Pr(Win|\{P_o(r_0), P_1(r_1)\})$ corresponds to the $r_0 \times r_1$ element of the $M \times N$ matrix.

We quickly discover that many implementations of particular ratio value combinations of ratio Strategy for Player 0 and ratio strategy for Player 1 do not terminate or terminate infrequently with extremely protracted runtime and amassed units. Essentially, certain ratio values for a player are too defensive and thus attack very little. When matched with a player who acts comparably, both players simple stockpile units until either one has enough units to pursue an attack or the game does not end as both players continue to only draft units. In the latter case, we simply state the game does not terminate. In the former case, it is possible to contrive games in which the player(s) will eventually attack. This scenario happens after drafting thousands, frequently tens of thousands of troops, one player arrives at a position to attack; this is very rare. We display one possible turn of a game in such a scenario in Figure 3.3. The game features Player 0 using $r_0 = 20$ and $r_1 = 5$. The ratio values were chosen to illustrate the point and, as we will see, are part of a much larger portion of non-terminating ratio combination.

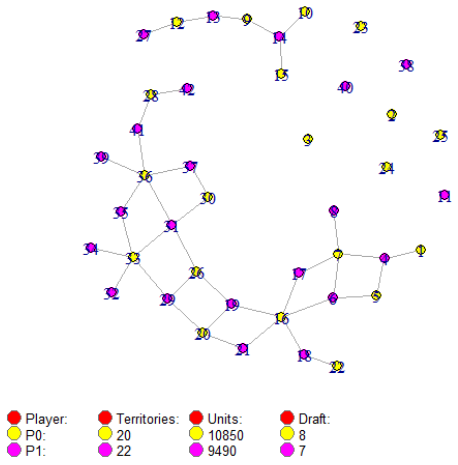

Figure 3.3: Example of Non-Terminating Game

We have no interest in continuing to model this unrealistic and impractical scenario as actual players would likely never be this defensive simultaneously. Instead, we choose to adjust our ratio strategy in a manner consistent with prior analysis in producing favorable results. We define a game to be *non-terminating* if at least one player has greater than $1,000$ units on the board at any given time. The purpose of this definition is straightforward, we do not aim to consider unrealistic games or games which do not provide sufficent endtimes for our given model. For games that do not terminate, we define the probability of success to be zero for both Player 0 and Player 1. Assuming a probability of success of zero for those instances provides the surface plot of probabilities in Figure 3.4.

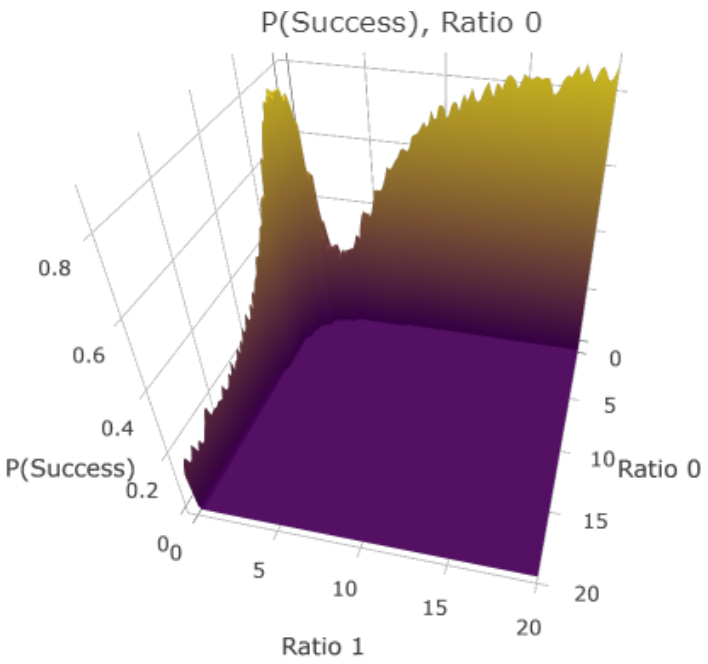

Figure 3.4: Ratio vs Ratio, Surface Plot

Recall the Ratio Strategy vs. Naive Strategy plot previously produced in Figure 3.1. Given that the naive strategy is essentially a case of the ratio strategy with $r = 0$, consider the above plot where $r_1 = 0$ and r_0 takes on the entire range of values. This cross-section of the surface plot is precisely the Ratio Strategy vs. Naive Strategy plot. Conversely, setting $r_0 = 0$ and letting r_1 take on the range of values provides the probabilistic complement of the curve at each point. Essentially, where Player 0 performs well, Player 1 performs poorly. Notice also that the vast majority of ratio value combinations produce probabilities of 0. There are various ratio combinations which produce terminating games, though we see all of these games feature relatively small ratio values.

3.4 The Piecewise Ratio Strategy

As detailed in the previous section, many games with particular ratio value combinations of ratio strategy for Player 0 and ratio strategy for Player 1 do not terminate. Thus, we define a new, more realistic strategy to ensure terminating games when crossing similar player strategies. We define the *Piecewise Ratio* value for Player i, with input ratio r_i to be:

$$r_i^* = \begin{cases} 0 & \text{if } A \geq D \text{ and } A \geq 5 \\ r_i & \text{otherwise} \end{cases} \tag{3.4}$$

The objective here is to place some boundary on *sufficient* success. For skirmishes up to size 10×10, we reproduce the table of expected attacker success probabilities. Our definition corresponds to initiating skirmishes in which the probabilities in Table 3.5 are in bold.

Table 3.5: Markov Chain Skirmish Results

A \ D	1	2	3	4	5	6	7	8	9	10
1	0.42	0.11	0.03	0.01	0.00	0.00	0.00	0.00	0.00	0.00
2	0.75	0.36	0.20	0.09	0.05	0.02	0.01	0.00	0.00	0.00
3	0.92	0.66	0.47	0.32	0.21	0.13	0.08	0.05	0.03	0.02
4	0.97	0.78	0.64	0.48	0.36	0.25	0.18	0.12	0.09	0.06
5	**0.99**	**0.89**	**0.77**	**0.64**	**0.50**	0.40	0.30	0.23	0.16	0.12
6	**1.00**	**0.93**	**0.86**	**0.74**	**0.64**	**0.52**	0.42	0.33	0.26	0.19
7	**1.00**	**0.97**	**0.91**	**0.83**	**0.74**	**0.64**	**0.54**	0.45	0.36	0.29
8	**1.00**	**0.98**	**0.95**	**0.89**	**0.82**	**0.73**	**0.64**	**0.54**	0.46	0.38
9	**1.00**	**0.99**	**0.97**	**0.93**	**0.87**	**0.81**	**0.73**	**0.65**	**0.56**	0.48
10	**1.00**	**0.99**	**0.98**	**0.95**	**0.92**	**0.86**	**0.80**	**0.73**	**0.65**	**0.57**

This approach forces players to attack once sufficient attacking units are amassed. Here, we allow a player to attack when the possible attacking units are the input ratio by the player, or set the ratio to zero when we have both $A \geq D$ and $A \geq 5$, which

corresponds to initiating skirmishes in which the probability of success is at least 0.5. The objective of the piecewise ratio strategy is to essentially compensate for strategies which in and of themselves are too defensive in nature, but don't want to sacrifice the gains of being defensive, namely, not losing skirmishes they initiate. This approach resolves our stalemate issues found with specific implementations of the ratio strategy versus the ratio strategy. We'll first examine the improvement of the piecewise ratio strategy against the naive strategy.

3.4.1 Piecewise Ratio Strategy vs. Naive Strategy

We've examined the naive player against the naive player, as well as the naive player against the ratio player, now we examine the naive player against the piecewise ratio player to examine any performance increase. The plot of the data is displayed in Figure 3.5.

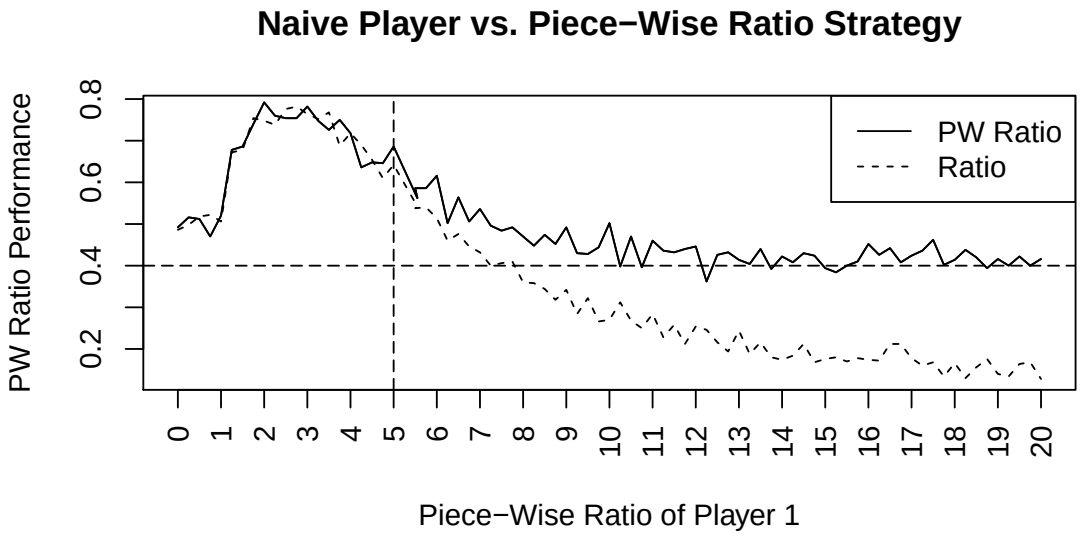

Figure 3.5: Piecewise Ratio Strategy Performance

We've included the data from the ratio strategy versus the naive strategy analysis

in the plot for ease of reference. Compared to the ratio strategy versus the naive strategy, the piecewise ratio strategy performs similarly to the ratio strategy on the interval $[0, 5]$. However, on the interval $(5, 20]$ we see a dramatic increase in performance as the ratio value for piecewise ratio increases. While the performance is not spectacular, we do see a substantial increase from the ratio strategy to approximately 15% to approximately 40% for the piecewise ratio strategy. The implementation of the piecewise ratio strategy against the naive player essentially allows a player to behave much more defensively than otherwise would be allowed without being substantially outperformed. We will next examine the piecewise ratio strategy against another player using the piecewise ratio strategy.

3.4.2 Piecewise Ratio Strategy vs. Piecewise Ratio Strategy

The next logical step in examining the piecewise ratio strategy is to consider the piecewise ratio strategy implemented against another piecewise ratio strategy. We assume ratio values of r between 0 and 20 by increments of 0.25 as done previously. The surface plot below displays the results of the simulations featuring two players implementing the piecewise ratio strategy.

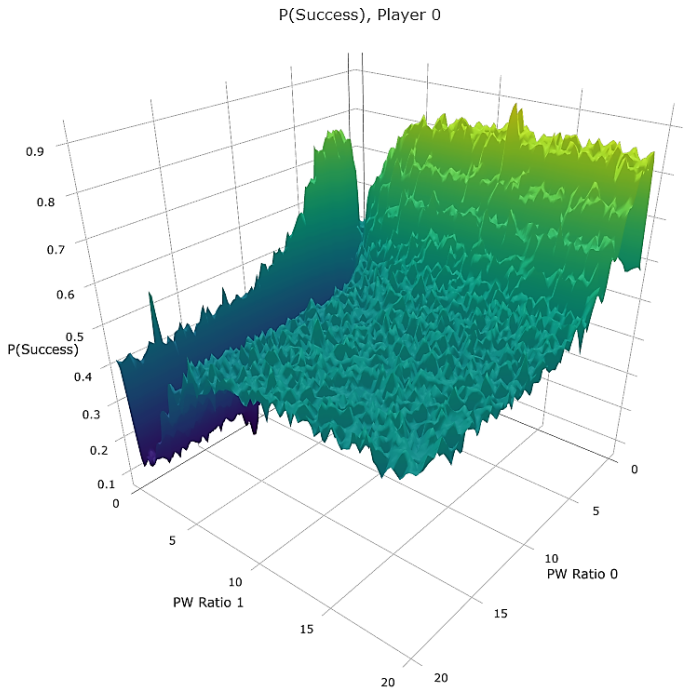

Figure 3.6: Piecewise Ratio vs. Piecewise Ratio

The general landscape of the surface plot indicates a large area of roughly 50% win percentages for either player. This area is located on intersections of large ratio values for each player. Decreasing the ratio value for either player leads to more interesting areas of the plot, which occur at areas close to the axes. We'll discuss those areas more thoroughly with the aid of a contour plot of the data as well.

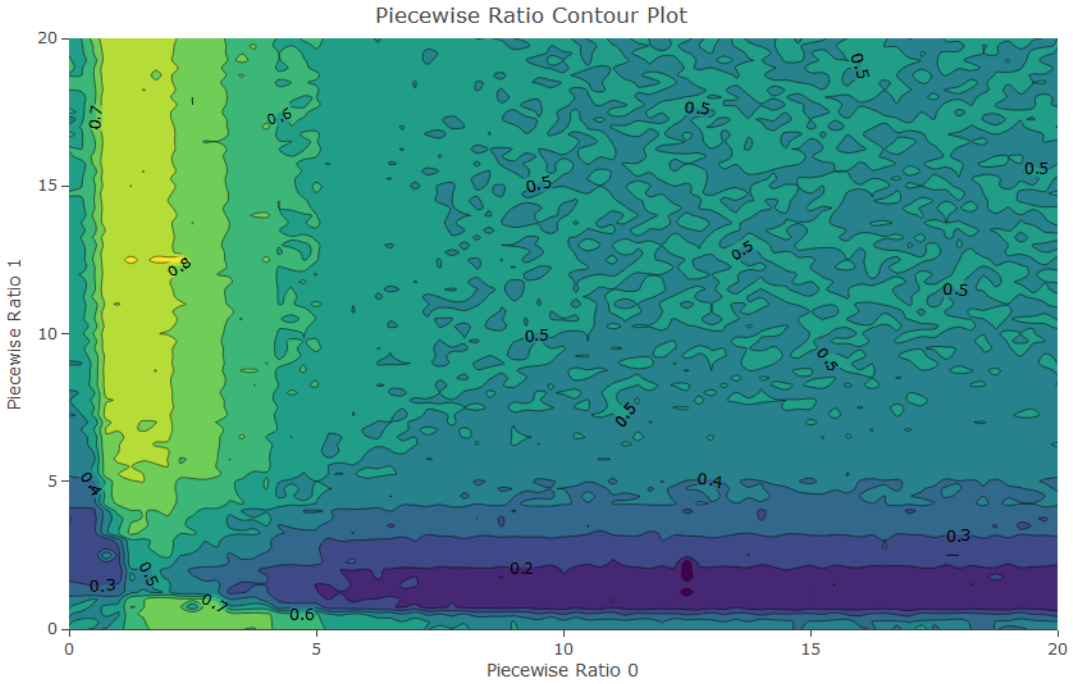

Figure 3.7: Piecewise Ratio vs. Piecewise Ratio

From the findings on the piecewise ratio strategy against the naive strategy, we have already found that when the piecewise ratio strategy assumes a ratio of 0, we simply produce a naive player.

The origin of the contour plot reproduces that result. Here we have two players adopting the piecewise ratio strategy with ratio of 0. This results in our previously found result of both players winning approximately 50% of the games. When the piecewise ratio for Player 1 is fixed to be zero (i.e. along the traditional "x-axis") and the piecewise ratio is allowed to scale between 0 and 20, the performance of piecewise ratio strategy for Player 0 is roughly half and half for the two players at near zero ratio values, then proceeds to spike for Player 0 near a ratio expected from the optimized ratio strategy results (in the [1.5,5] range), and ultimately levels off to roughly 50% win for either player. Notice, for large ratio values for Player 1 the win percentage is still roughly 50%. Compared to previous findings with the ratio strategy, this is

a drastic improvement from performance continuing to diminish at the ratio value increased.

Let's consider now when piecewise ratio strategy for Player 0 maintains a ratio of 0 and the piecewise ratio strategy for Player 1 is allowed to scale between 0 and 20 (i.e., along the "y-axis"). As previously found, areas near the origin produces win probabilities near 50% for either player. An increase in the ratio for Player 1 into the area found to be most successful in the optimized ratio section shows a sharp decrease in win probability for Player 0. This is to be expected, as the probability of win in this area for Player 1 is the same as the Probability for loss for Player 1 in the same area along the piecewise ratio 0 axis. Accordingly, further increases in ratio values for Player 1 level off the performance for both players to around 50%. Again, this change in performance from the ratio strategy findings is drastic, where we found high ratio values inevitably led to very low win percentages; here we have produced a modest 50% win probability.

The areas where the ratios for either player are in the range considered in the optimized ratio section are perhaps most interesting. With the ratio for Player 1 to be variable between 0 and 20, we see Player 0 attains a win probability between 70% and 80% for ratios above 1 and neighboring 2.5. Conversely, if we consider Player 0 ratios to be variable between 0 and 20, we see a win probability of merely $20-30\%$ for Player 1 ratio values in the discussed range between 1 and slightly above 2.5. Reference to the surface plot indicates just how drastic these win probabilities are in context of two identical players opposing one another.

The remaining section of the contour plot, the areas where both players take on relatively high ratio values (e.g., 5 and above) shows win probability for each player of about 50%. By design, this result is not surprising. The piecewise ratio strategy was designed to produce a player who acts aggressively when possessing sufficient

units. We are not surprised then to see two players who only attack when amassing large amounts of units win with roughly equal proportion. This corresponds to both players acting defensively until the piecewise ratio criterion is met, then acting as *Naive*, producing the win probabilities we have already found.

Overall, the surface and contour plots inform us that aggressive strategies are met well with low, non-zero ratio values. The use of a *Naive Strategy* essentially performs well only when the other player acts very defensively, which can be remedied by implementing an addition criterion as we have with the piecewise ratio strategy. Further, if both players act defensively, the games essentially end by the toss of a coin. And finally, if a player adopts the piecewise ratio strategy with a ratio previously shown to be successful, the player will continue to be successful against both overly aggressive and overly defensive players. The piecewise ratio strategy seems to "fix" defensive players to at least the point of producing a challenge, and produces favorable results when implemented with low ratio values as well.

3.4.3 Piecewise Ratio Strategy with Continents

The next step in determing the effectiveness of the piecewise ratio strategy is to perform a series of continent simulations. For this set of simulations we choose to use two players implementing the piecewise ratio strategy with different ratios. We choose $r_0^* = 2.5$ and $r_1^* = 10$. The value for r_0^* is chosen as we have repeatedly shown this ratio to be effective against both aggressive and defensive players. The value of r_1^* is chosen as it is a reasonably defensive player, one which would be ineffective in the ratio strategy setting, which will serve well to examine the effects of giving it control of continents.

Thus, we proceed to allow the piecewise ratio player with $r_1^* = 10$ to have control of each continent in separate simulations. The results of the simulations are displayed

in the table below.

Table 3.6: PW Ratio at $r = 2.5$ vs. PW Ratio at $r = 10$ with Given Continent

Strategy \ Continent	None	Asia	N. America	Australia	S. America	Africa	Europe
Piece-Wise Ratio at $r = 2.5$	0.752	0.504	0.426	0.482	0.450	0.329	0.276
Piece-Wise Ratio at $r = 10$	0.248	0.496	0.574	0.518	0.550	0.671	0.724
Difference	0.504	0.008	-0.148	-0.036	-0.100	-0.342	-0.448

The results of these simulations are perhaps most interesting. The implementation of the piecewise ratio strategy with $r_1^* = 10$ performed very well. In the case of the Simulations featuring Asia, we find near equal win percentages. However, for all other continents, the piecewise ratio player with $r_1^* = 10$ has outperformed the piecewise ratio player with $r_0^* = 2.5$. What we have found is that even though Player 1 is very defensive, it is an effective strategy when given continents. It is not unrealistic to imagine a scenario in which a player intentionally gains early control of a continent and proceeds to play defensively under the piecewise ratio strategy guidelines and outperforms otherwise strong performing strategies.

3.5 Strategy Comparison & Conclusions

Concluding our discussion of strategies, we have defined a variety of heuristic methods of play. The most basic of these heuristics is the naive strategy, which amounts to a player attacking as much as possible with as much force as possible. We found that two players against one another both implementing the naive strategy results in roughly equal win percentages for each player. This result was precisely as suspected. We then allowed one of the aforementioned players to have a single continent at the beginning of the game; an assumption to simply examine the impact of continent control. The player who was given each of the continents over the various simulations for each continent won with varying increased performance depending on the continent given. We somewhat surprisingly found the possession of Asia did not strongly increase the

win percentage, whereas control of Europe drastically increased the performance of the player. Control of other continents altered performance to somewhere in between those percentages. We found that while the naive strategy made poor decisions, early control of a continent, to varying degrees, would compensate for this effect.

To explore a more realistic heuristic method of play, we introduced the ratio strategy. Reframing the naive strategy to account for both the number of attackers and defender, we view the naive strategy as essentially requiring $A \geq 0 * D$ for number of attackers, A, and number of defenders D. The ratio strategy is based on this reframing of the naive strategy and allows the 0 in the previous inequality to scale over any value of interest. We explored values in the range of $[0, 20]$ by increments of 0.25. We posed the ratio strategy against a player implementing the naive strategy for each of the ratio values and found a clear trend. For low ratio values, the ratio player and naive player performed similarly. This was expected, as a low ratio corresponded roughly to same strategy as the naive strategy. However, increasing the ratio modestly, towards the range of $[1, 5]$ proved to be a drastic improvement in win percentage. This range roughly corresponds to a player not being overly aggressive, as the naive strategy is designed to be, but more modest in attacking. Beyond the range of $[1, 5]$, we found that the probability of winning for the ratio player dropped off substantially. As the values approached the end of our range of consideration, we found that the win percentage continued to decrease. While we only observed ratios as high as 20, we reasonably suspect the win percentages will continue to trend downward. The reasoning here is simple, the higher ratio values essentially direct a player to attack only under conditions unlikely to be met in realistic play. Furthermore, while the ratio player with a high ratio attempts to continue stockpiling units, the naive player will continually attack and deplete the enemy units. While the naive player will have very little in reserve, the ratio player will likely be unable to amass any substantial amount of units and will slowly be taken over.

From the simulations featuring the naive strategy against the ratio strategy, we did find one area of ratio of particular interest; the moderate ratio values in which the ratio strategy performed well. We then examined the interval of $[0.95, 4.5]$ more closely by increasing our simulation count to produce less error and refining our interval into increment of 0.05. We found that values approximately between $[1.5, 3]$ all performed very similarly, with nearly 80% win percentages. We subsequently denoted any ratio strategy using a ratio of 2.5 to be the optimized ratio strategy.

Given the optimized ratio strategy, we became interested in finding the impact of giving continents to a naive player against a optimized ratio player. The reasoning here is that previous findings regarding the continents have shown that possession of a continents strongly influences the ability to win a game, even when strategically performing poorly. The results of the optimized ratio strategy against the naive strategy with continents showed that the optimized ratio strategy still outperformed the naive strategy, with the exception of the naive player being given Europe.

The next test we posed to the ratio strategy was to examine how the ratio strategy performed against another player implementing the ratio strategy. For small ratio values, we found results consistent with previous findings. However, we quickly discovered that many ratio combinations put both players in a far too defensive position. For ratios in the range of $[0, 20]$ for the two players, the vast majority of games simply do not terminate; both players simply stockpile units indefinitely. Since we had every desire to put a threshold on when the appropriate time to attack is, we then modified the ratio strategy by introducing the piecewise ratio strategy.

The piecewise ratio strategy amounts to implementing the ratio strategy when the opponent has a large mass of units or the player has few units to attack. Essentially, when the count of attacking units is low, the piecewise ratio strategy acts in accordance with some ratio as prescribed by the ratio strategy. Once the piecewise ratio player

has a enough units with respect to the enemy, the piecewise ratio player is then designed to attack more in line with the criterion for the naive strategy. That is, when the piecewise ratio player has sufficient units, attacks should be implemented more liberally than required by the ratio strategy.

To examine the performance of the piecewise ratio strategy we performed a similar set of exercises. We first examined the performance of the piecewise ratio strategy against the naive strategy. Specifically, this performance is referenced against the performance of the ratio strategy against the naive strategy. A similar curve is produced which shows the piecewise ratio strategy outperformed the ratio strategy. Further, the objective of the piecewise ratio strategy is met. The piecewise ratio strategy increases the performance of ratios which are very high; namely, defensive players have a much higher probability of winning against a naive player than we previously found.

To further examine the piecewise ratio strategy, we examined the piecewise ratio strategy against another player implementing the piecewise ratio strategy, as was our aim when examining the ratio strategy against the ratio strategy. These simulations proceeded successfully for ratio values between 0 and 20. The simulation results were then used to produce a surface plot to understand the general landscape of these interactions. We found several interesting results. For low ratio values, we found results in line with our previous general findings regarding the ratio strategy and the naive strategy. For values near the ratios examined in the optimized ratio strategy, we found similar performances, which for the piecewise ratio strategy also produced optimal results. The more interesting results are found when both players implements high ratios. The win percentages were roughly 50% for a wide range of high ratios, ratios deemed “too defensive” in the ratio strategy. Previously, games featuring two players of similar ratios would fail to terminate.

We then chose two ratio values, the first value $r_0^* = 2.5$ and the second value $r_1^* = 10$, to perform a series of continent-oriented simulations. Our desire was to determine if the *traditionally* defensive ratio of $r_1^* = 10$ could be shown to be successful against a proven ratio value of $r_0^* = 2.5$ by simply allowing for onset continent control. The results of these simulations were surprising. When no continent was given, the piecewise ratio player with $r_0^* = 2.5$ strongly outperformed the piecewise ratio player with $r_1^* = 10$ as would be expected. However, when given a continent we found that the piecewise ratio player with $r_1^* = 10$ outperformed the piecewise ratio player with $r_0^* = 2.5$ in most instances. Several of the simulations showed Player 1 strongly outperforming Player 0. It appears the piecewise ratio strategy is a successful bridge between playing defensively and playing moderately.

Chapter 4

Analysis of Simulation Data

4.1 A Brief Introduction to Neural Networks

4.1.1 Scope and History

Based on our simulations of various strategies, a natural pursuit is to then build a predictive model for the outcome of a game. To achieve this, we build neural network models using the graphical predictors and game predictors previously discussed. We start with a review of the relevant neural network concepts and terminologies. Neural networks make up a broad class of statistical models. Originally motivated by modeling biological neurons, McCulloch and Pitts (1943) gave the neural network model

$$n_i(t) = H(\sum_{j \to i} w_{ji} n_j(t-1) - \theta_i) \tag{4.1}$$

for the sum over j neurons which are connected to some neuron i with transformation function H given by the threshold function $H(x) = I(x > 0)$. The output $n_i(t)$ is the signal produced from neuron i at a time t where the neurons have associated

weights in which $0 < w_{ij} < 1$. Present day neural networks represent a broad class of models, which are commonly visually represented by a network architecture of three distinct types of layers (Lesmeister, 2015). Neural network models in which information is passed forward from one layer to the next are referred to as *feed-forward neural networks*. We will limit our discussion to networks of this variety.

Specifically, neural network models are non-linear regression models and are sufficiently flexible to approximate any smooth function with an arbitrary degree of accuracy (Venables & Ripley, 2002). Furthermore, the implementation of a neural network model requires no underlying assumptions of the structure of our data (Lesmeister, 2015).

4.1.2 Architecture and The Forward Pass

While neural networks are generally represented as featuring three distinct types of layers of nodes, in the input layer to the hidden layer(s) to the output layer, a neural network is actually a composition of functions $f(\mathbf{x}; \mathcal{W})$ of a vector of inputs $\mathbf{x}$ and collection of weights $\mathcal{W}$ visually organized by layers and nodes (Efron & Hastie, 2016). One such visualization of a general neural network featuring 3 input nodes corresponding to the labels "Var1", "Var2", and "Var3", 4 hidden layer nodes, and 1 output node corresponding to the label "Response1" is provided in Figure 4.1. Visually, the network representation of neural networks is a powerful aid, though much detail is lost in relying on this alone for understanding (Fritsch & Guenther, 2016). We will address both the architectural scheme of neural networks as well as the function representation, beginning with the former by examining Figure 4.1.

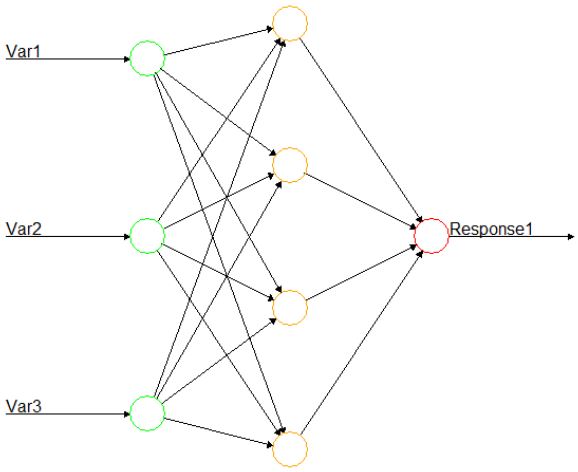

Figure 4.1: Neural Network Example

The edges between the nodes are assigned numeric values and are referred to as *weights*. We will refer to the collection of these weights as $\mathcal{W}$. Each portion of the neural network architecture performs a specific task which will be detailed below.

The first layer in a neural network model is the *input layer*, which corresponds to the green nodes in Figure 4.1. The input layer is made up of a series of input nodes which represent the indepedent variables of interest in some analysis. These nodes represent the "predictors" in a more classical statistical analysis such as simple linear regression. We typically denote these inputs as a vector with components x_i. The nodes in the input layer feature weighted connections to nodes in the following layer(Lesmeister, 2015).

The layer following the input layer is denoted as the *hidden layer* and features a user-specified number of hidden nodes. The size of the hidden layer as well as the magnitude of the weights are frequently established by experimentation or through

some separate algorithm. A frequent criticism of neural networks is the ambiguity of both the behavior of the hidden layer and the choice of the size of the hidden layer. Both of these issues can be addressed satisfactorily for many instances of analysis(Lesmeister, 2015). Neural network models are not suitable for all types of analysis. Much effort in modern statistics and computer science is devoted to determining the appropriate weights and sizes of neural networks. While we will restrict our discussion to a single hidden layer, any number of hidden layers can be implemented.

The weight w_{ih}^{l} indicates the l^{th} weight from the input layer(i) to the hidden layer(h). To proceed from input layer with an input x_i with weighted connection w_{ih}^{l} to the hidden layer, we multiply the value of x_i by the weight w_{ih}^{l} to produce a numeric value $x_i \times w_i^{(1)}$. This is repeated for all weights connected to nodes in the input layer. Additionally, neural networks feature *bias* terms at each layer beyond the input layer, which we have produced in Figure 4.2. The bias terms, α_l for bias at layer l, act as an additional input for the layer which is analogous to an intercept term in traditional linear models. The bias nodes and weights are colored blue in Figure 4.2.

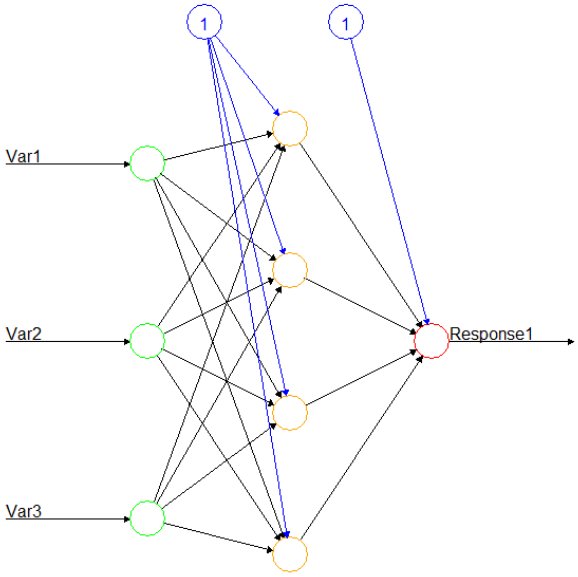

Figure 4.2: Neural Network Example with Bias

Note also from Figure 4.2 that the bias nodes have a value of 1 inside of them. The bias nodes always receive an input of 1. Additionally, as with all inputs the bias nodes have an associated weight, α_l for the l layer. Thus, to proceed from the bias node at the input layer, α_h, to the hidden layer, we multiple the value of the input, 1, by the weight to produce a value $1 \times \alpha_h$. Once all inputs and weights have been appropriately multiplied, the hidden layer node with terminating arrows takes all of the values produced from each weight and sums the values. Once all values have been summed at each hidden node, individually, the sum at the hidden layer node undergoes a transformation by a prespecified function ϕ_h, termed the *activation function*. Generally, the transition from the input layer to the hidden layer is given by

$$\phi_h(\alpha_h + \sum_l w_{ih}^{(l)} x_i) \tag{4.2}$$

Common activation functions include the sigmoid function, the logistic function, the linear function, the hyperbolic tangent function, and the rectified linear function(Lesmeister, 2015). The choice of activation function varies on the scope of the problem, as well as an element of industry *rules of thumb*. Typically, some variant of the logistic function is used for classfication and some variant of the linear function is used for a linear output. While we will assume each neuron uses the same activation fuction, it is important to note that each neuron beyond the input layer has it's own activation function and these functions can differ at each layer and indeed each node within each layer.

There is also a historic element to the choice of the activation function as well. George Cybenko first showed feed-forward neural networks containing finite hidden nodes could approximate any continuous function under limited assumptions. This is a result known as the *Universal Approximation Theorem* (Cybenko, 1989). The activation function shown by Cybenko was the sigmoid function, though this result was repeated for a multitude of other activation functions by later researchers.

Once all values at the hidden layer have been put through the activation function, the values produced from the hidden layer nodes serve as inputs for the next layer. The next and final transition of interest in the neural network architecture is the transition from the hidden layer to the output later. As represented by Figure 4.1 and Figure 4.2, the transition is represented from traversing the four orange nodes to the single red node of the output layer. Depending in the type of analysis, this output layer can produce a classification or approximation to a curve given inputs $\mathbf{x}$.

The values entering the output layer undergo precisely the same procedure as processing values from the input layer to the hidden later. The output layer receives values by multiplying the values exiting the hidden layer and weight associated with that particular connection. This is repeated for each connection terminating on an

output layer node. The incoming values on the output layer nodes are then summed, and become the inputs for an activation function associated with the output layer node of interest. Given Equation 4.2 and the procedure to evaluations from the hidden layer to output layer, we can produce the following composition of functions

$$y_k = \phi_o(\alpha_k + \sum_l w_{hk}^{(l)} \phi_h(\alpha_h + \sum_l w_{ih}^{(l)} x_i)) \tag{4.3}$$

for output values y_k, output activation function ϕ_o, hidden layer activation function ϕ_h, output layer bias α_k, hidden layer bias α_h, hidden layer to output layer weights $w_{hk}^{(l)}$, input layer to hidden layer weights $w_{ih}^{(l)}$, and input values x_i (Venables & Ripley, 2002). Note, Equation 4.3 is given to produce an output y_k, while in general a neural network is a composition of functions, $f(\mathbf{x}; \mathcal{W})$, which takes on the form given in Equation 4.3. Also note that the summations at both *layers* are indexed over the l weights contained between the two layers of interest typically do not have the same number of weights. For example, in Figure 4.2 we have $l = 16$ weighted edges between the input layer and hidden layer while only $l = 5$ edges between the hidden layer and output layer.

The procedure of passing inputs through the input layer, applying appropriate weights and transformations until the output layer is referred to as the *feed-forward* part of the neural network algorithm (Lesmeister, 2015). While our discussion of neural network architecture is centered around describing an already existing network, it is important to note than the values of all the weights in $\mathcal{W}$ are typically the result of some optimization algorithm used to adjust the values of the weights to reduce error.

4.1.3 The Backpropagation Algorithm

Once the forward pass of the neural network is completed, an optimization method for reducing error is implemented. The most common procedure for reducing error is the *backward propagation of errors(backpropagation) algorithm.* The *backpropagation* algorithm assesses the error associated with the training data after the feed-forward portion of the algorithm (Lesmeister, 2015). The error is calculated by a user-defined loss function, L; this function is typically taken to be the sum of squared error given by

$$L = \sum_i L_i[y_i, f(x_i)] = \sum_i (y_i - f(x_i))^2 \tag{4.4}$$

for data with true response y_i and vector of inputs x_i (Lesmeister, 2015). Since the weights chosen at the beginning of the algorithm are initialized to small random values, the errors found at this stage in the procedure tend to be large. Then reducing the error in the n samples in the training set $\{x_i, y_i\}_1^n$ of inputs x_i and responses y_i can be viewed as an optimization problem in which we solve

$$\min_{\mathcal{W}} \left\{ \frac{1}{n} \sum_{i=1}^{n} L[y_i, f(x_i)] \right\} \tag{4.5}$$

for neural network outputs $f(x_i)$ and target values y_i.

Naturally, to minimize the loss function we differentiate $L[y, f(\mathbf{x}, \mathcal{W})]$ with respect to any weight in $\mathcal{W}$ to observe the change in error with respect to small changes in weights. Since $f(\mathbf{x}, \mathcal{W})$ is a composition of functions, the derivative will be found by implementing the chain rule over the series of compositions (Efron & Hastie, 2016). Most methods for minimizing the above expression implement gradient descent (Efron & Hastie, 2016). A wide variety of alterations and modifications exist. We will

omit the details of computing the error changes with respect to weight changes as we will not implement unmodified backpropagation in our analysis, though the spirit of the algorithm is maintained. A common pursuit in reducing the error is to further modify the fit criterion by introducing a *regularization* term to Equation 4.5.

4.1.3.1 Weight Decay

Perhaps the most common regularization term is to penalize the square of the weight parameters. This application of L2 regularization is termed *weight decay* in neural networks and is simply modifying the fit criterion to

$$L + \lambda \mathcal{J}(\mathcal{W}) \tag{4.6}$$

where $\mathcal{J}(\mathcal{W})$ is some regularization function which is often taken to be the sum of squares of the weights of $\mathcal{W}$ and $\lambda \geq 0$ is the weight decay parameter. The addition of the weight decay term has the interpretation of penalizing large components of the collection of weights more heavily. The use of weight decay aids the optimization process as well as protects against overfitting (Venables & Ripley, 2002). By applying our loss function 4.4, we now have the minimization problem given by

$$\min_{\mathcal{W}} \left\{ \frac{1}{n} \sum_{i=1}^{n} L[y_i, f(x_i)] + \lambda \mathcal{J}(\mathcal{W}) \right\} \tag{4.7}$$

where $\mathcal{J}(\mathcal{W})$ is given explicitly by

$$\mathcal{J}(\mathcal{W}) = \frac{1}{2}|\mathbf{w}|^{\mathbf{2}} \tag{4.8}$$

where $\mathbf{w}$ is a vector of weights in $\mathcal{W}$ and $|\mathbf{w}|$ is the L2 norm of the form

$$\sqrt{w_1^2 + w_2^2 + ... + w_n^2} \tag{4.9}$$

The factor of 1/2 on $\mathcal{J}(\mathcal{W})$ is typically included to simply aid in the calculation of the gradient (Venables & Ripley, 2002). Given 4.5, 4.7, and 4.8, we have

$$\min_{\mathcal{W}} \left\{ \frac{1}{n} \sum_{i=1}^{n} L[y_i, f(x_i)] + \lambda \frac{1}{2} |\mathbf{w}|^2 \right\} \tag{4.10}$$

Loss functions are typically not convex in the elements of $\mathcal{W}$, so solving this minimization problem is generally an unrealistic task (Efron & Hastie, 2016). Given that, we typically aim to find a sufficient local optima of our desired loss function as shown in Equation 4.11

$$\min_{\mathcal{W}} \left\{ \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \frac{1}{2} |\mathbf{w}|^2 \right\} \tag{4.11}$$

Minimizing the error is essentially adjusting weights until a better fit is found. These adjustments are based on optimizing the error functions with respect to changes in the weights.

4.1.3.1.1 BFGS Algorithm

Weight decay can be implemented by a variety of optimization algorithms, the *BFGS algorithm* which was simultaneously published by Broyden (1970), Fletcher (1970), Goldfarb (1970), and Shanno (1970) is a popular method for implementing weight decay. The BFGS algorithm provides an approximation to the inverse Hessian matrix which aids in adjusting weight changes as given in Equation 4.12, instead of the gradient of the error function alone as done with unmodified backpropagation. The BFGS method belongs to the class of quasi-Newton methods which are based on

Newton's method and (in this context) perform the following iteration to achieve a local minima

$$\mathbf{w}_{n+1} \leftarrow \mathbf{w}_n - [HL(\mathbf{w}_n)]^{-1} \nabla L(\mathbf{w}_n) \tag{4.12}$$

where $\mathbf{w}_{n+1}$ is a weight vector of non-bias weights which depend on the computation of the inverse Hessian matrix $[HL(\mathbf{w}_n)]^{-1}$, a square matrix of second-order partial derivatives of the loss function, and $\nabla L(x)$, the gradient vector of the loss function. The procedure aims to provide weights which at each iteration of the algorithm reduce the estimate of the loss function. The utilization of the Hessian matrix allows for an efficient update in descending the gradient as the local curvature of the loss function is described more efficiently than in first order techniques. The computation of the inverse Hessian matrix is frequently impractical for large scale problems as explicit calculations become expensive in both time and space. To remedy this problem, Quasi-Newton methods have been developed with the aim of approximating the inverse Hessian in Equation 4.12 (Nocedal & Wright, 2006). Among these methods is the BFGS method which iteratively constructs an approximation to the Hessian matrix. The steps for the algorithm are presented below.

The BFGS method requires a starting value $x^{(0)}$ and an estimate for the matrix of second-order partial derivative of the loss function, $\nabla^2 L(x^{(0)})$, which is typically taken to be the appropriate size identity matrix at the beginning of the algorithm. Then for $k \in \mathbb{N}$, perform the following iteration to produce an approximation of the Hessian matrix:

Find the search direction p_k by

$$p_k = -H_k^{-1} \nabla L(x_k) \tag{4.13}$$

Determine step length α_k in direction of p_k by

$$\alpha_k = arg\min L(x_k + \alpha p_k) \tag{4.14}$$

We define the vector

$$s_k = \alpha_k p_k \tag{4.15}$$

The new iterate of x becomes

$$x^{(k+1)} = x^{(k)} + s^{(k)} \tag{4.16}$$

We define

$$y^{(k)} = \nabla L(x^{(k+1)}) - \nabla L(x^{(k)}) \tag{4.17}$$

Then Hessian approximation at $k+1$ is given by

$$H_{k+1} = H_k - \frac{H_k s^{(k)} (s^{(k)})^T H_k}{(s^{(k)})^T H_k s^{(k)}} + \frac{y^{(k)} (y^{(k)})^T}{(y^{(k)})^T s^{(k)}} \tag{4.18}$$

Then the inverse Hessian approximation is given by

$$\begin{aligned} H_{k+1}^{-1} = H_k^{-1} &- \frac{H_k^{-1} y^{(k)} (s^{(k)})^T + s^{(k)} (y^{(k)})^T H_k^{-1}}{(s^{(k)})^T y^{(k)}} \\ &+ \frac{((s^{(k)})^T y^{(k)} + (y^{(k)})^T H_k^{-1} y^{(k)})(s^{(k)} (s^{(k)})^T)}{((s^{(k)})^T y^{(k)})^2} \end{aligned} \tag{4.19}$$

Thus, at each iteration the BFGS algorithm gives an update for the approximation

of the inverse Hessian, which is then used to perform an update on the weights (Nocedal & Wright, 2006). One interation of the forward pass and backpropagation constitutes one *epoch*. The processes of the feed-forward pass and backpropagation are carried out either over some number of epochs or until some tolerance of error is met (Lesmeister, 2015).

4.2 Neural Network Analysis of Data

4.2.1 Model Structure

Given the discussion of neural network architecture and optimization, we proceed to build a neural network to examine data extracted from simulations of the game *RISK*. Our aim is to build a neural network model to perform the task of supervised learning of classifying games as a *0* or *1*, indicating a win for *Player 0* or *Player 1*, respectively. The activation function we use for our classification problem is the sigmoid function given by:

$$\phi(x) = \frac{1}{1 + e^{-x}} \tag{4.20}$$

A plot of the sigmoid function given in Equation 4.20 is displayed in Figure 4.3 which indicates the classifications given to neural network output values.

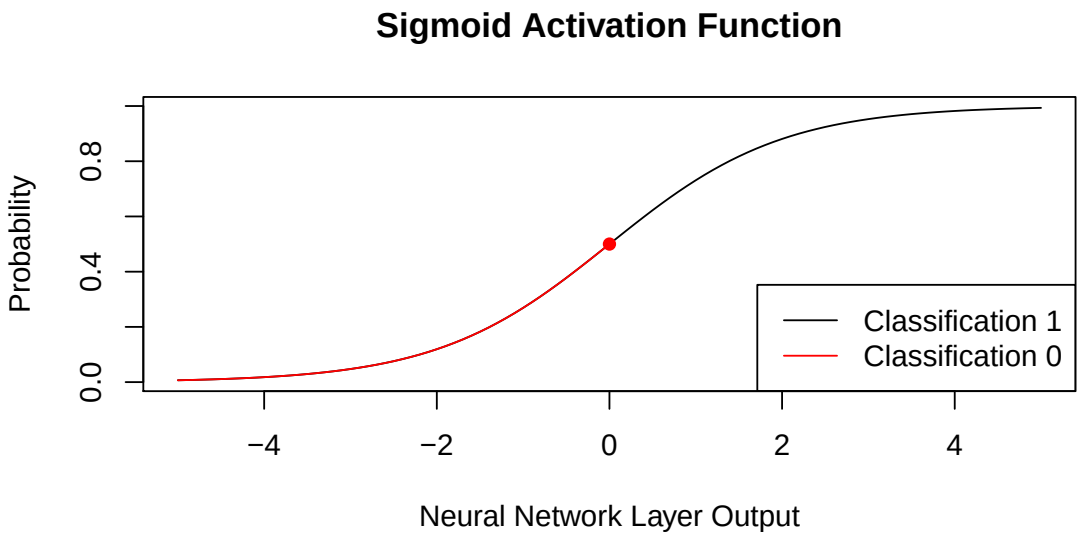

Figure 4.3: Sigmoid Function

The sigmoid function serves our purpose nicely as it turns a classification problem into placing values on a smooth curve. Furthermore, the sigmoid is differentiable, which aids in optimization, and is bounded from $[0, 1]$, which makes interpreting the network outputs as probabilities to be straightforward.

We build single layer feed-forward neural network models on a dataset featuring 5000 simulations; 2500 simulations from each of *Naive Strategy* versus *Naive Strategy* games and *Optimized Ratio Strategy* versus *Naive Strategy* games. The simulations feature all of the previously discussed 28 predictors belonging to both the class of game predictors and the class of graphical predictors. The objective of our analysis is to provide a classification, namely identifying the winning player (Lesmeister, 2015). For our analysis, we will implement a single hidden layer neural network of the form

$$y_k = \phi_o \left(\alpha_k + \sum_l w_{hk}^{(l)} \phi_h (\alpha_h + \sum_l w_{ih}^{(l)} x_i) \right) \tag{4.21}$$

4.2.2 Fitting Models via Grid Search

Given the computationally intensive tasks of producing a forward pass, backpropagation with weight decay through the BFGS method, and making predictions after fitting, we implement software to fit our model. Such software is provided in the *nnet* package in R. The *nnet* package makes use of neural networks of the variety that we have thus far described (Venables & Ripley, 2002).

To build our model on the data set of 5000 simulations we perform three separate procedures simultaneously to ensure we have produced a model which fits the data well. First, models are built using a portion of the 5000 simulations as a training set and the remainder of the simulations used as a test set. We use training set sizes of 5% to 100% of the 5000 simulations. That is, we build models on 20 different training/test set splits. The objective here is to examine the performance of the model after being built. A model which performs well will produce similar classification results for the test set as well as the training set. Furthermore, models which are given a large portion of the data for training tend to be less general and simply remember the data.

Secondly, to assure performance desires are met, we implement a grid search on the size of the single hidden layer and the weight decay parameter λ (Jed Wing et al., 2017). We allow the weight decay to take on values of 0 to 1.5 by increments of 0.1 and we allow the hidden layer to contain 1, 2, or 3 nodes. The restriction on number of nodes serves two purposes. First, we want to minimize model complexity given our abundance of predictors. While a model with greater than 3 nodes in the hidden layer may be more flexible, we restrict our hidden layer to a maximum size of 3. Second, we regard our desire to prevent overfitting as more important than allowing additional, perhaps unhelpful, model flexibility through increased hidden layer size. Furthermore, since weight decay is our primary means of preventing overfitting, we provide our grid search with much more granular values of λ. The values we allow

our models to take on are 0 to 1.5, by increments of 0.1. Since we are performing a grid search of hidden layer size and weight decay value, the more granular increments of weight decay increases the number of models to be built and tested substantially. Restricting the size of the hidden layer to 3 helps to minimizes the number of models to be trained and examined.

Thus, for each training data split, we build 48 neural networks models, one for each grid point on our 3×16 grid for 3 possible hidden layer nodes and 16 possible weight decay values. For each of these 48 models, we pick the model with the highest accuracy on the entire data set as the *best* model under the given parameter values. The table in Figure 4.1 represents the models built on various training percentages of the data set by highest accuracy. The table displays the size of the hidden layer, the decay parameter value, and the accuracy on the full data set.

Table 4.1: Model Results

Training Percent	Size	Decay	Accuracy
0.05	3	1.1	0.9784
0.10	**1**	**1.0**	**1.0000**
0.15	3	1.0	0.9914
0.20	3	1.0	0.9964
0.25	2	0.7	0.9994
0.30	3	0.7	0.9970
0.35	2	1.3	0.9990
0.40	3	1.1	0.9998
0.45	3	1.2	0.9998
0.50	3	1.2	0.9988
0.55	3	0.8	1.0000
0.60	3	1.3	0.9960
0.65	3	1.0	0.9992
0.70	3	1.2	0.9984
0.75	3	0.6	0.9992
0.80	3	1.0	0.9990
0.85	3	0.7	0.9988
0.90	2	0.8	1.0000
0.95	2	0.9	1.0000
1.00	2	1.2	0.9992

Clearly, the model results indicate that virtually all of the models perform similarly. Additionally, there are models of hidden layer size 1, 2, and 3 and decay values between 0.6 and 1.3. In general, producing a model which performs well and is relatively simple in comparison to similarly performing models is desirable. Given this preference for a simpler model that performs well, the model built on 10% of the training data performs quite well. The model has a single node in the hidden layer and is therefore is as simple of an architecture as possible in our grid search. The corresponding decay value of 1.0 is reasonably modest and presents no challenges in computation. While all of the models perform very well in terms of accuracy, the model built on 10% of the training data correctly classifies 100% of the 5000 simulations.

Thusly, we have a resulting architecture of 28 input nodes, 1 hidden layer node in the single hidden layer, and 1 output node in our neural network. Furthermore, since we have $l = 28$ nodes in the input layers and $l = 1$ node in the hidden layer, we have

$$y_k = \phi_o \left(\alpha_k + w_{h,k} \phi_1 (\alpha_h + \sum_{l=1}^{28} w_{i,h}^{(l)} x_i) \right) \tag{4.22}$$

for an output value y_k, output activation function ϕ_o, hidden layer bias α_h, hidden layer activation function ϕ_1, output layer bias α_k, weight of inputs $w_{i,h}$ going to hidden layer 1 from input value x_i, and the single weight from the hidden layer to ourput layer $w_{h,k}$. This equation can be visually representation by the neural network architecture in Figure 4.4.

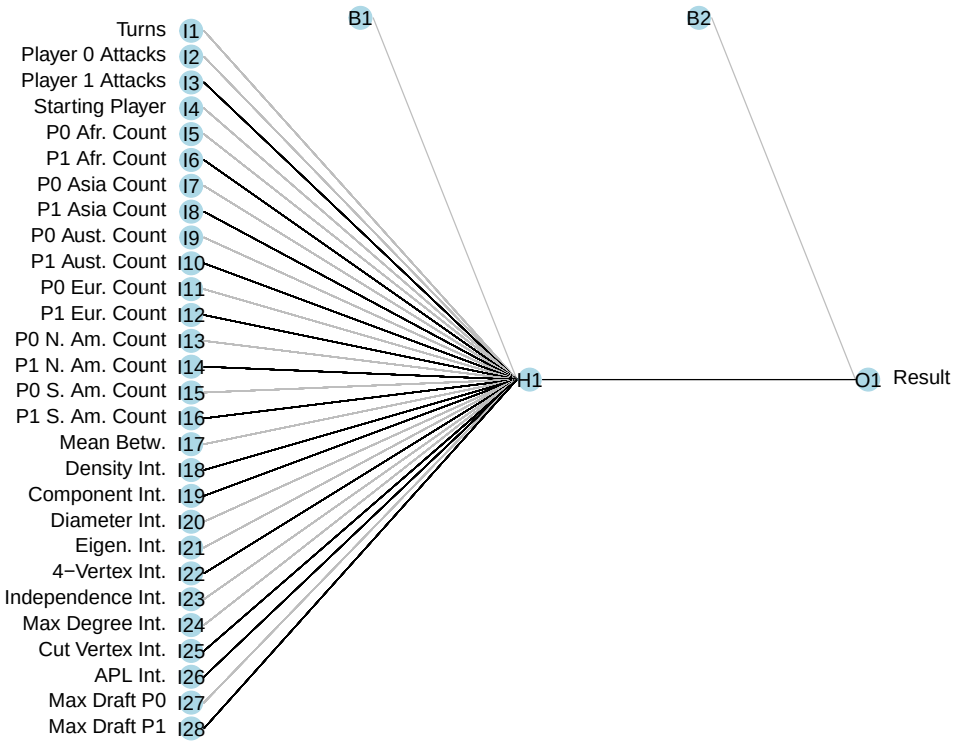

Figure 4.4: Neural Network of Model

While the visualization is helpful in understanding the sequence of events and overall strcture, the equation of the neural network is perhaps more important. The equation for the neural network produced in 4.4 is given in Equation 4.23.

$$
\begin{aligned}
y = \phi_0(&-2.8845567981+5.6122363192\phi_1(\\
&-0.02967278 * Attacks_{P0}+0.1297937 * Attacks_{P1}\\
&-0.01885228 * Starter+ - 0.04633782 * Turns\\
&-0.05951199 * Africa_{P0}+0.06862268 * Africa_{P1}\\
&-0.04970139 * Asia_{P0}+0.0393970464 * Asia_{P1}\\
&-0.1292855218 * Australia_{P0}+0.0835975635 * Australia_{P1}\\
&-0.0633787972 * Europe_{P0}+0.0895343945 * Europe_{P1}\\
&-0.0402959702 * N.America_{P0}+0.0483062897 * N.America_{P1}\\
&-0.0718316674 * S.America_{P0}+0.0478625126 * S.America_{P1}\\
&-0.0072990364 * \mu(Betweenness)+0.0030263006 * \int GraphDensity\\
&0.0032703752 * \int Cluster+ - 0.0066139075 * \int Diameter\\
&-0.0452837284 * \int EigenCentrality+0.0010148911 * \int RiskGraphs\\
&-0.0013735898 * \int IndependenceNumber+ - 0.0190105750 * \int DegreeMax\\
&0.0032267072 * \int CutVertices+0.0163418510 * \int AveragePathLength\\
&-0.3603233040 * DraftMax_{P0}+0.3044527454 * DraftMax_{P1}\\
&- 0.0004512878))
\end{aligned}
\tag{4.23}
$$

where ϕ_o and ϕ_1 are both the sigmoid function. For a complete understanding we include an example of a computation for predicting the probability of a Player 0 success given data from a single simulated game.

Table 4.2: Model Prediction by Hand

Inputs		Weight	Hidden Layer Input	Hidden Layer Output	Weight	Output Layer Input	Output Layer Output
Turns	6	-0.04633782					
P0 Attacks	3	-0.02967278					
P1 Attacks	37	0.1297937					
Starter	1	-0.01885228					
P0 Africa	0	-0.05951199					
P1 Africa	1	0.06862268					
P0 Asia	0	-0.04970139					
P1 Asia	0	0.03939705					
P0 Aust	0	-0.1292855					
P1 Aust	4	0.08359756					
P0 Eur	0	-0.0633788					
P1 Eur	4	0.08953439					
P0 N. Am.	0	-0.04029597					
P1 N. Am.	0	0.04830629					
P0 S. Am.	0	-0.07183167	7.549033	0.999473	5.612236	5.609282	0.93847
P1 S. Am.	2	0.04786251					
Mean Betw.	3.762	-0.007299036					
GD Int.	0.738	0.003026301					
Comp. Int	1387.5	0.003270375					
Diam. Int	230	-0.006613907					
Eigen. Int.	92.6	-0.04528373					
4-Vert. Int	802.2	0.001014891					
Ind. Int	1655	-0.00137359					
MaxDeg Int	127	-0.01901058					
CV Int	280	0.003226707					
APL Int	101.8	0.01634185					
P0 Max D.	7	-0.3603233					
P1 Max D.	24	0.3044527					
Bias 1	1	-0.0004512878					
Bias 2	1					-2.884557	

The rightmost column of Table 4.2 indicates a classification of 1, since $0.93847 > 0.5$. This classification indicates that given the data set, our model predicts Player 1 won the game.

Chapter 5

Discussion

We began our analysis by reproducing the results of Tan(1997) and Osborne(2003) in Markov chain approximation to skirmishes. The Markov chain modeling of the skirmishes in *RISK* represents a large stepping stone for further analysis. An adequate understanding of the events of a skirmish are pivotal in developing a deeper understanding of the entire game. We took the results from the Markov chain approximations as reinforcements for our intuition and later as an aid in developing the piecewise ratio strategy. Upon completion of the Markov Chain approximations we introduced our model of the game.

To model the game, we made several simplifying assumptions. A player which abides by all of the assumptions in Assumptions for Naive Analysis was termed a naive player and more generally represented the naive strategy. The most modified assumption was the rate at which a player would attack. The naive player represented a player which would attack whenever possible with as much force as possible. We made a series of changes to this assumptions and found an increase in performance with each change.

To implement these assumptions we built a discrete time network model as

outlined in The Network Model. The aim of the model was to combine graph theoretic properties and properties of the game to extract features of *RISK* which could be used as predictors in a statistical analysis. Obviously, collecting enough empirical data from a *model* implemented by two human players interacting would be problematic. We chose to built a program to make use of our assumptions and provide output corresponding to each graphical change at the completion of a game.

To collect such data, we modeled the *RISK* board as a dynamic graph in which edges undergo deletion and insertion. Specifically, we deleted edges when vertices of the same color shared an edge and reinstated edges when vertices were of opposite color and such an edge existed in the underlying graph. This representation of the game allowed for the end of the game to be seen as an empty graph on 42 vertices. Additionally, this graphical representation of the game allows the graphical properties to exhibit a discernable trend.

This depiction of the game allowed for the examination of many features, as discussed in The Unit List and Graphical Properties of G_t. Our model produced a matrix associated with each t^{th} player attack which could be graphically analyzed. We explored a wide range of graphical predictors in our model, as discussed in Graphical Properties of G_t. We found that our collection of predictors, both graphical and game-based, served well in our statistical pursuit.

The next major step in implementing our model was to examine a series of strategies for our players to use. We begin with the naive strategy as outlined in The Naive Strategy. We quickly found that as designed, the naive strategy suffered from a host of flaws. Namely, the naive strategy seemed to be an overly aggressive player. An additional step we took at each introduction of a strategy was to perform a series of simulations in which a player was given onset control of a continent. For the naive strategy, the objective of these simulations was to simply examine how the

naive strategy performed when given more resources and a better starting position in the game. Of course, the player given continent control at the onset performed better and we generally found that the naive player was too aggressive. We found that possession of Africa or Europe increased performance the most, and provided reasoning for each of those cases.

To remedy this, we introduced a generalized strategy based on the initial attacking criterion of the naive strategy which we called the ratio strategy. The ratio strategy featuring a scalable attacking criterion that depended on the number of attacking armies and the number of defending armies. This is in contrast with the naive strategy which required only that there be non-zero units available for attack. Through examining the interactions between the naive player and the ratio player we found an interval of ratio values in which the ratio player strongly outperformed the naive player. However, we found for small ratio values that the ratio strategy performed similarly to the naive strategy. This finding was not particularly surprising as small ratios correspond to strategies very similar to the naive strategy.

However, we also found that the performance of the ratio strategy at large ratio values was poor. Specifically, for sufficiently large ratios we found the naive strategy strongly outperformed the ratio strategy. This scenario corresponds to a ratio player being far too defensive and allowing the naive player to constantly chip away the units and territories belonging to the ratio player.

The interval in which the ratio strategy outperformed the naive strategy was thoroughly examined and we took a ratio values of $r = 2.5$ to be the optimal ratio in the ratio strategy; as noted "optimal" is used in the sense of a maximum value on a single variable curve, as opposed to optimal as defined in game theory. The win percentage of this ratio against a naive player was found to be roughly 80%. We then took this specific ratio of the ratio strategy to use against a naive player which is

given onset control of continents. We found that only when given Europe did the naive strategy outperform the ratio strategy with a ratio of $r = 2.5$.

These strategic findings indicate that the adoption of a moderate strategy is generally indicative of successful play. While many other strategy definitions could exist, and certainly without any respect to a ratio, we found that attacking with roughly 2.5 times the number of units the opponent has on a territory tends to produce favorable results. Of course, from the Markov chain findings there is a higher probability when attacking with much larger forces. The drawback in that interpretation is that attacking in those scenarios typically indicates the player has been amassing units and playing defensively for far too long in the game. These are the instances in which playing moderately is pivotal, where playing defensively will be most detrimental.

Given the successful performance of the ratio strategy under the constrained aforementioned situation, we sought to next examine the performance of a ratio player against another ratio player. The attempt at these simulations led to the finding that many games of this nature, for varying ratio values, simply do not terminate. Effectively, the combinations of ratios produce two players who are too defensive to attack in most circumstances. The non-terminating games presented yet another hurdle in developing a more successful strategy. To combat this problem, we introduced the the piecewise ratio strategy.

The piecewise ratio strategy served the purposed of resolving players acting too defensively by modifying the criterion of attack to be more realistic when a player has amassed a significant amount of units available for attack. While many thresholds could be set, we set the ratio for attacking to be a piece-wise function. First, a player will attack under any circumstances when $A \geq D$ and also $A \geq 5$. Otherwise, the player will attack in accordance with whatever pre-chosen ratio.

While the piecewise ratio strategy shows itself to be an improvement upon the previous strategies, other successful modifications likely exist. The definition of the piecewise ratio strategy relied on the probability of attacker success findings. Alternatively, ratios of simular structure to the piecewise ratio strategy could be based on the findings on the Expected Attacker Losses, in which we only commit to skirmishes in which our expected losses are within some desired range.

The first test of the piecewise ratio strategy was to examine the performance against a naive player. These simulations produced desirable results. Namely, the low ratios corresponded to essentially the same outcome as we found with the ratio strategy vs. the naive strategy. We also found similar performance around mid-range ratios. However, for large ratios we found a substantial increase in performance. While many of the high ratios had performances around 40%, this was a substantial increase over the roughly $10 - 15\%$ win percentages for the ratio strategy before the *piecewise* component was introduced.

The next test for the piecewise ratio strategy was to perform simulations in which the piecewise ratio strategy was posed against another player implementing the piecewise ratio strategy. This scenario was the intention of the ratio strategy vs. the ratio strategy in which we found many combinations of games to be non-terminating. From the simulations featuring the piecewise ratio strategy vs. the piecewise ratio strategy, we produced a surface plot of the results. The general theme of the results was that low ratios performed adequately, while intermediate ratio values performed strongly, and high ratio values performed reasonably well for their defensive nature.

From the change in strategies from the ratio strategy to the piecewise ratio strategy, the biggest improvement was found to be with the defensive players. These players essentially became offensive enough to be competitive, but maintained the edge of defensiveness of not losing skirmishes frequently. To further examine the

performance of defensive players under the piecewise ratio strategy, we performed a series of simulations featuring a piecewise ratio strategy player with $r = 2.5$ and another piecewise ratio strategy player with $r = 10$ which was given onset control of continents. Our base of the two players interacting without continents showed that the player with $r = 2.5$ strongly outperformed the other player.

When given the continent, however, the piecewise ratio strategy player with $r = 10$ performed equally well or made at least a 10% improvement in overall win percentage, including an improvement of up to nearly 45% with Europe. These simulations were at least somewhat surprising. We have produced a strategy that is reasonably adaptive and realistic, and also allows the player to perform well under various circumstances. As desired, we produced a simple heuristic for play, with our most complicated strategy effectively hinging on two requirements and producing win percentages around 80% when paired with players using similar strategies. Perhaps surprisingly, the findings of the piecewise ratio strategy indicated that a mix of luck in holding continents and playing defensively is *not* a bad strategy. In general, we find that defensive players should be more offensive and offensive players should be slightly more defensive.

While our model did accomplish the classification task for which it was built, certain limitations were introduced to make the project feasible. For one, we neglected to implement the *RISK* cards at any point in our analysis (a feature which can swing the outcome of the game quickly). Additionally, the important simplifying assumption of two players is only mildly realistic. Furthermore, our player strategies did not feature any sort of *preference* for territories. Any number of preferences could be assigned based on our graphical findings. For example, a strategy which aims to surround the opponent could be implemented. In real game play, a player is likely to try to capture the remaining portions of a continent or deny an opponent such an

opportunity. Also, in reality when a player holds a continent they will probably try to protect it more than other areas of their control. We did not require this of our strategies and doing so would likely increase the performance of continent-holding players. While we could produce endless options to enhance our strategy definitions, our current findings produce a valuable *baseline* for future work.

Another area in which our strategies could be improved is to apply some preference toward the order in which enemy territories are attacked. That is to say, we left open the possibility for a player to commit to an attack with a force and that force subsequently become stuck in an area surrounded by territories already held by the same player(i.e. attacking your way into a corner of your own territories). A more efficient player would attack with only as many units as necessary or attack territories in a sequence that would leave the remaining force on the *front line*. This problem could also be remedied by allowing players to make use of the *reinforcement phase*, which would allow players to move units forward in the event that units became isolated from enemy territories.

Another, perhaps more interesting possible modification would be to design players to deliberately alter the draft number of the opposing player. For example, if the enemy has 21 territories, then they draft 7 units. The other player could be programmed to attack with purely the aim of reducing the territories of the enemy to 20, then the enemy would only draft 6 units. A player strategy could be designed to reduce enemy draft numbers in this fashion. An enemy could also do the same thing for an opponent holding a continent. This is a realistic adjustment to make for a player strategy. In all, there are many such modifications which could be made. This is entirely reflective of the nature of human behavior in games which feature a wide variety of options.

Moreover, we could further generalize our notions of strategies to include players

acting with different strategies at different portions of the game. The strategies we have designed allow a player to act a certain way for the entire game. An adaptive element could prove helpful. For example, perhaps a player decides that playing aggressively or defensively at the onset of the game somehow impacts performance positively. Also realistically, perhaps a player realizes their loss is imminent and commits to a Hail Mary series of attacks in an attempt to turn the game in their favor, as opposed to waiting idly while the other player continues to outperform.

Given the variety and breadth of data we can collect from the model, the next logical step in our analysis is to perform a modeling task on the data to either understand the influence of the predictors from an explanatory perspective or to use the predictors to build a predictive model. We opt to produce a predictive model with the main method for analyzing data being a feed-forward single layer neural network with weight decay. The decision to use such a model was based on several factors. Foremost, the type of predictors we used was diverse. Many predictors we used were count variables which potentially poses a problem to certain generalized linear models. While extensions of generalized linear models exist to handle the use of count predictors, we were less confident in their implementation due to relatively little literature in this area.

Moreover, in many instances a number of the graphical predictors were highly correlated with one another; this usually presents a problem in statistical modeling. Typically, a neural network model handles such collinearity much more smoothly and requires little intervention from the user. With modern applications ranging in the millions for parameters, multicollinearity is often an assumed property of the set of predictors. Moreover, the results of our analysis suggest the multicollinearity did not have any noticeable adverse effects.

The statistical model built on data collected from our network model performed

very well in classifying the outcome of games post hoc. While the main objective of this analysis was to determine which player would win given the state of the predictors at the end of the game, our model could be implemented on a turn-by-turn basis. We could extract the predictors from our model at each time t, instead of cumulatively at the end of a simulation, and use the predictor data available thus far in the game to make a prediction on the outcome of the game. This would amount to producing "in-game" probabilities. Given the neural network model for making predictions, we would likely have trouble in interpreting which action would be best to make to produce an optimal win percentage. However, this dilemma could be resolved by performing simulations on each of the player's available moves and acting in accordance with the one that produces the highest win percentage. Overall, a wide range of possible statistical excursions exist from the basis of our findings.

Appendix

Code for the simulation of *RISK* games can be found at

http://campus.murraystate.edu/academic/faculty/cmecklin/MunsonThesis/JacobMunsonThesisCode.html.

The code represents a minimal reproduction of the contained work. The file "RISK_Simulation.R" contains a wrapper function for implementing all code found in "RISK_game_files.R". Both files will need to be executed entirely to used the simulations wrapper function.

References

Broyden, C. G. (1970). The convergence of a class of double-rank minimization algorithms 1. General considerations. *IMA Journal of Applied Mathematics*, *6*(1), 76–90. https://doi.org/10.1093/imamat/6.1.76

Corporation, A. M. (n.d.). Retrieved October 14, 2017, from https://ucarecdn.com/6803914a-1715-4c38-8bd1-7cb9a150ac6b/

Csardi, G., & Nepusz, T. (2006). The igraph software package for complex network research. *InterJournal*, *Complex Systems*, 1695. Retrieved from http://igraph.org

Curran, J. M. (2013). *Bolstad2: Bolstad functions*. Retrieved from https://CRAN.R-project.org/package=Bolstad2

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems (MCSS)*, *2*(4), 303–314. https://doi.org/10.1007/BF02551274

Efron, B., & Hastie, T. (2016). *Computer age statistical inference: Algorithms, evidence, and data science* (1st ed.). New York, NY, USA: Cambridge University Press.

Fletcher, R. (1970). A new approach to variable metric algorithms. *The*

Computer Journal, *13*(3), 317–322. https://doi.org/10.1093/comjnl/13.3.317

Fritsch, S., & Guenther, F. (2016). *Neuralnet: Training of neural networks*. Retrieved from https://CRAN.R-project.org/package=neuralnet

Gelman, A., & Hill, J. (2007). *Data analysis using regression and multilevel/hierarchical models*. Cambridge, United Kingdom: Cambridge University Press.

Goldfarb, D. (1970). A family of variable-metric methods derived by variational means. *Mathematics of Computation*, *24*(109), 23–26. Retrieved from http://www.jstor.org/stable/2004873

Gross, J. L., Yellen, J., & Zhang, P. (2013). *Handbook of graph theory* (2nd ed.). United Kingdom: Chapman & Hall/CRC.

Jed Wing, M. K. C. from, Weston, S., Williams, A., Keefer, C., Engelhardt, A., Cooper, T., ... Hunt., T. (2017). *Caret: Classification and regression training*. Retrieved from https://CRAN.R-project.org/package=caret

Kolaczyk, E., & Csárdi, G. (2014). *Statistical analysis of network data with R*. New York: Springer.

Lesmeister, C. (2015). *Mastering machine learning with R*. Birmingham, United Kingdom: Packt Publishing. Retrieved from https://books.google.com/books?id=8yLEjgEACAAJ

McCulloch, W., & Pitts, W. (1943). A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, *5*, 115–133.

Nocedal, J., & Wright, S. J. (2006). *Numerical optimization* (2nd ed.). Singapore: World Scientific.

Osborne, J. A. (2003). Markov chains for the Risk board game revisited. *Mathematics Magazine*, *76*(2), 129–135. Retrieved from http://www.jstor.org/stable/

3219306

Shanno, D. F. (1970). Conditioning of quasi-newton methods for function minimization. *Mathematics of Computation*, *24*(111), 647–656. Retrieved from http://www.jstor.org/stable/2004840

Tan, B. (1997). Markov chains and the RISK board game. *Mathematics Magazine*, *70*(5), 349–357. Retrieved from http://www.jstor.org/stable/2691171

Venables, W. N., & Ripley, B. D. (2002). *Modern applied statistics with S* (4th ed.). New York: Springer. Retrieved from http://www.stats.ox.ac.uk/pub/MASS4